



*ByteOps.swe@gmail.com*

## Norme di progetto

### Informazioni documento

---

<b>Redattori</b>	A. Barutta R. Smanio N. Preto
<b>Verificatori</b>	E. Hysa L. Skenderi D. Diotto
<b>Amministratore</b>	F. Pozza
<b>Destinatari</b>	T. Vardanega R. Cardin

# Registro delle modifiche

Versione	Data	Autore	Verificatore	Dettaglio
0.3.0	21/11/2023	A. Barutta	E. Hysa	Sezione Gestione della configurazione
0.2.0	06/11/2023	A. Barutta N. Preto	R. Smanio	Sezione Comunicazione, Gestione incontri
0.1.0	05/11/2023	A. Barutta N. Preto	R. Smanio	Sezione Documentazione, Introduzione

# Indice

ByteOps

## Contents

<b>1</b>	<b>Introduzione</b>	<b>6</b>
1.1	Finalità del documento . . . . .	6
1.2	Glossario . . . . .	6
1.3	Riferimenti . . . . .	6
1.3.1	Riferimenti normativi . . . . .	6
1.3.2	Riferimenti informativi . . . . .	6
<b>2</b>	<b>Processi di supporto</b>	<b>7</b>
2.1	Documentazione . . . . .	7
2.1.1	Introduzione . . . . .	7
2.1.1.1	Documentation as Code . . . . .	7
2.1.2	Ciclo di vita dei documenti . . . . .	8
2.1.2.1	I redattori . . . . .	9
2.1.2.2	I verificatori . . . . .	11
2.1.2.3	Il responsabile . . . . .	11
2.1.2.4	L'amministratore . . . . .	11
2.1.3	Struttura del documento . . . . .	11
2.1.3.1	Verbali: struttura generale . . . . .	12
2.1.4	Regole tipografiche . . . . .	13
2.1.5	Abbreviazioni . . . . .	14
2.1.6	Strumenti . . . . .	15

2.2	Verifica . . . . .	15
2.2.1	Verifica dei documenti . . . . .	15
2.2.1.1	I verificatori . . . . .	15
2.3	Gestione della configurazione . . . . .	16
2.3.1	Introduzione . . . . .	16
2.3.2	Numeri di versionamento . . . . .	16
2.3.3	Tecnologie . . . . .	17
2.3.4	Repository . . . . .	17
2.3.4.1	Struttura repository . . . . .	17
2.3.4.2	Sincronizzazione e Branching . . . . .	18
2.3.5	Controllo di configurazione . . . . .	18
2.3.5.1	Change request (Richiesta di modifica) . . . . .	18
2.3.6	Configuration Status Accounting (Contabilità dello Stato di Configurazione) . . . . .	19
2.3.7	Release management and delivery . . . . .	20
<b>3</b>	<b>Processi organizzativi</b>	<b>20</b>
3.1	Pianificazione . . . . .	20
3.1.1	Assegnazione dei ruoli . . . . .	20
3.1.1.1	Responsabile . . . . .	21
3.1.1.2	Amministratore . . . . .	21
3.1.1.3	Analista . . . . .	21
3.1.1.4	Progettista . . . . .	22
3.1.1.5	Programmatore . . . . .	22
3.1.1.6	Verificatore . . . . .	22
3.1.1.7	Ticketing . . . . .	23
3.2	Comunicazione . . . . .	24
3.2.1	Comunicazioni sincrone . . . . .	24
3.2.2	Comunicazioni asincrone . . . . .	24
3.2.2.1	comunicazioni asincrone interne . . . . .	24
3.2.2.2	comunicazioni asincrone esterne . . . . .	24
3.3	Gestione degli incontri . . . . .	25

3.3.1	Incontri formali esterni . . . . .	25
3.3.2	Incontri formali interni . . . . .	25
3.3.2.1	Meeting settimanali interni . . . . .	25

# 1 Introduzione

## 1.1 Finalità del documento

L'obiettivo fondamentale del seguente documento è quello di stabilire le linee guida e le *best practice* che ciascun membro del gruppo deve seguire per garantire un approccio efficiente ed efficace nel processo di realizzazione del progetto didattico.

I processi e le relative attività contenute nel seguente documento sono state definite a partire dallo Standard ISO/IEC 12207:1995.

Il documento è strutturato secondo i processi del ciclo di vita del software e presenta una gerarchia in cui ogni processo si configura come una serie di attività. Ciascuna attività è composta da procedure dotate di obiettivi, scopi e strumenti ben definiti.

In aggiunta, il documento dettaglia le convenzioni relative all'utilizzo dei diversi strumenti adottati durante lo sviluppo del prodotto.

È importante sottolineare che questo documento è in continua evoluzione poiché le norme definite al suo interno vengono regolarmente riesaminate, aggiornate e ottimizzate seguendo un approccio incrementale.

## 1.2 Glossario

Nella documentazione è incluso il **Glossario**, dove vengono definiti tutti i termini specifici o potenzialmente ambigui presenti nei vari documenti correlati al progetto. La presenza di una nota a pedice con la lettera G accanto a un termine indica che è possibile trovare la sua definizione nel **Glossario**.

## 1.3 Riferimenti

### 1.3.1 Riferimenti normativi

- **Standard ISO/IEC 12207:1995**: Standard internazionale che definisce un modello di ciclo di vita del software e in cui sono definite delle linee guida per la gestione dei processi software e le relative attività.  
In sintesi, stabilisce una struttura per organizzare le diverse fasi e le attività coinvolte nel ciclo di vita del software, aiutando le organizzazioni a sviluppare prodotti software in modo più efficiente ed efficace.

### 1.3.2 Riferimenti informativi

- **Documentazione Git**

- [Documentazione Github](#)
- [Documentazione L<sup>A</sup>T<sub>E</sub>X](#)
- "Clean Code" di Robert C. Martin

## 2 Processi di supporto

### 2.1 Documentazione

#### 2.1.1 Introduzione

La documentazione costituisce l'insieme di informazioni scritte che accompagnano un prodotto software, fornendo dettagli utili a sviluppatori, distributori e utenti.

Tra gli scopi della documentazione troviamo:

- Comprensione del prodotto senza supporto umano, usandola come mezzo di comunicazione;
- Segnare il confine tra creatività e disciplina;
- Assicurare che i processi produttivi si svolgano con la qualità attesa.

L'obiettivo della sezione è:

- Definire delle procedure ripetibili che permettano di uniformare la documentazione prodotta dal gruppo ed il metodo di lavoro;
- Raccogliere e organizzare le norme che i membri del team devono seguire così da semplificare l'operazione di scrittura dei documenti.

Tali norme dovranno essere applicate da tutti i membri del team ed i sorgenti L<sup>A</sup>T<sub>E</sub>X che contengono tale documentazione verranno inseriti nel repository disponibile all'indirizzo:

<https://github.com/ByteOps-swe/Sorgente-documenti>

##### 2.1.1.1 Documentation as Code

L'approccio che si intende adottare è quello di "Documentation as Code" (Documentazione come Codice) che consiste nel trattare la documentazione di un progetto software allo stesso modo in cui si tratta il codice sorgente. Questo approccio è incentrato sull'utilizzo di pratiche e strumenti tipici dello sviluppo software per creare, gestire e distribuire la documentazione.

Alcuni aspetti chiave di "Documentation as Code" includono:

- **Versionamento**

- **Scrittura in Formato Testuale**
- **Automazione**
- **Collaborazione**
- **Integrazione Continua**
- **Distribuzione**

Questo approccio porta diversi vantaggi, tra cui una maggiore coerenza, una migliore tracciabilità delle modifiche e facilità di manutenzione. Inoltre, il concetto di "Documentation as Code" si allinea con la filosofia DevOps, dove la collaborazione e l'automazione sono valori chiave.

## 2.1.2 Ciclo di vita dei documenti

Il ciclo di vita dei documenti è una sequenza di stati e attività:

### 1. **Stato: Necessità**

- (a) Nasce la necessità di una documentazione, per obbligo o per opportunità;
- (b) Pianificazione della sua stesura;
- (c) Suddivisione in sezioni;
- (d) Durante le riunioni, si procede alla discussione e alla definizione collettiva di una traccia per il contenuto;
- (e) Assegnazione delle sezioni ai redattori tramite task su ITS.

### 2. **Stato: Redazione**

- (a) Ogni tipo di documento viene creato secondo la struttura specificata nei prossimi paragrafi;
- (b) Il team realizza il documento redigendone il contenuto rispettando le norme definite;

### 3. **Stato: Verifica**

- (a) Quando completato il documento viene revisionato dai verificatori;
- (b) Il documento viene compilato e il PDF generato viene inserito nella repository all'indirizzo: <https://github.com/ByteOps-swe/Documents> ;

### 4. **Stato: Manutenzione**



- (a) Manutenzione: ogni documento sotto configuration management deve essere modificato in accordo alle norme di versionamento e di change management.
- (b) La richiesta di modifica nasce da una nuova necessità sul contenuto del documento e riporta il documento allo stato omonimo.

### 2.1.2.1 I redattori

Il redattore responsabile della redazione di un documento o di una sua sezione deve seguire lo stesso approccio richiesto per la codifica del software, adottando il workflow noto come "feature branch".

#### Caso redazione nuovo documento, sezione o modifica

Dalla repository "Sorgente documenti" contenente i sorgenti  $\text{\LaTeX}$ , il redattore dovrà creare un nuovo branch Git in locale e passare ad esso mediante l'utilizzo del comando:

```
git checkout main
git checkout -b branch_identificativoTask
```

Il nome del branch destinato alla redazione o modifica del documento o di una sua sezione deve essere descrittivo al fine di consentire un'identificazione immediata e precisa del documento o della sezione su cui si sta lavorando. Pertanto, relativamente alla redazione dei documenti, deve essere adottata la seguente **convenzione per la nomenclatura dei branch**:

- Il nome del branch deve essere uguale al nome del documento senza estensioni, senza spazi e versione, scritto in camel case. (ex. VerbaleInterno14\_11\_2023) (Riferimento: 2.1.4);
- Nel caso della redazione di una sezione si pone alla fine "\_NomeSezione". (ex. NormeDiProgetto\_Documentazione);
- Nel caso di modifica di una sezione si pone alla fine "\_ModNomeSezione". (ex. NormeDiProgetto\_ModDocumentazione).

A questo punto, dopo aver creato il documento, la sezione o la modifica assegnata e avviato la stesura, affinché gli altri redattori possano continuare il lavoro, è necessario rendere il branch accessibile anche nella repository remota, seguendo i seguenti passaggi:

```
git add .
git commit -m "Descrizione del commit"
git push origin branch_identificativoTask
```

#### Caso modifica documento in stato di redazione

Qualora il redattore intenda continuare la stesura di un documento (o sezione) iniziato da un altro redattore, sono necessari i comandi:

```
git pull
git checkout branch_identificativoTask
```

### Salvataggio e condivisione progressi di task non completate

Alla fine di una sessione di modifiche di un file, nel caso si desideri rendere accessibile ai membri il lavoro non ancora completato e, pertanto, non pronto alla verifica, è necessario:

1. Eseguire il push<sub>G</sub> delle modifiche fatte nel branch

```
git add .
git commit -m "Descrizione del commit"
git push origin branch_identificativoDocumento
```

2. Nel caso di problemi con il punto 1:

```
git pull origin branch_identificativoDocumento
```

3. Risolvi i conflitti e ripeti punto 1.

### Completamento compito di redazione

Al termine del loro lavoro, i redattori:

1. Segnalano il completamento dell'attività a loro assegnata (essa sia la stesura completa di un documento o di una sua parte) posizionando l'issue relativa nella colonna "Da revisionare" della [DashBoard documentazione](#).
2. Attuano una **pull request**:
  - (a) Aggiornare il registro delle modifiche inserendo i dati richiesti in una nuova riga e incrementando la versione (Il verificatore è definito all'assegnazione della task, presente nella descrizione della Issue dell'ITS).
  - (b) Eseguire i passaggi dettagliati nel caso "**Salvataggio e condivisione progressi di task non completate**".
  - (c) Vai sul repository su GitHub » Apri la sezione "pull request" » Clicca "New pull request"
  - (d) Seleziona come branch di destinazione "main" e come branch sorgente il ramo utilizzato per la redazione del documento (o sezione) da validare
  - (e) Clicca "Create pull request"
  - (f) Dai un titolo e una breve descrizione alla pull request » Seleziona i verificatori su "Reviewers" » Clicca "Create pull request"

Se i verificatori non convalidano il documento (o sezione), i redattori riceveranno feedback allegati alla pull request relativi ai problemi identificati.

#### **2.1.2.2 I verificatori**

I compiti e le procedure dei verificatori sono dettagliate al paragrafo 2.2.1.1.

#### **2.1.2.3 Il responsabile**

Nel processo di redazione dei documenti, il compito del responsabile consiste nel:

- **Identificare i documenti o le sezioni da redigere.**
- **Stabilire le scadenze entro cui devono essere completati.**
- **Assegnare i redattori e i verificatori ai task.**
- **Approvazione:** Prima della conclusione del suo mandato, il responsabile si riserva il diritto di approvare il lavoro o richiedere eventuali ulteriori modifiche su tutti i documenti redatti e verificati nel periodo in cui ha esercitato la propria carica.

#### **2.1.2.4 L'amministratore**

Nel processo di redazione dei documenti, il compito dell'amministratore consiste nel:

- **Inserire nel ITS le attività specificate dal responsabile:**
  - Redattori: assegnatari della issue;
  - Verificatori: specificati nella descrizione della issue;
  - Scadenza.

### **2.1.3 Struttura del documento**

I documenti ufficiali seguono un rigoroso e uniforme schema strutturale, il quale richiede un'osservanza scrupolosa.

#### **Prima pagina**

Nella prima pagina di ogni documento deve essere presente:

- Nome e mail del gruppo
- Nome del documento
- Redattori
- Verificatori
- Destinatari

## Indice

Tutti i documenti generici devono essere provvisti di un indice dove saranno elencate le varie sezioni e sottosezioni, con la possibilità di raggiungerle direttamente tramite click. In caso di presenza di figure nel documento, sarà presente anche un indice relativo.

## Piè di pagina

In ogni piè di pagina deve essere presente:

- Nome del gruppo
- Nome del documento
- Numero di pagina

## Registro delle modifiche

Tutti i documenti devono essere provvisti di un registro delle modifiche in formato tabellare che contiene un riassunto delle modifiche apportate al documento nel corso del tempo. La tabella deve essere inserita nella sezione registro delle modifiche subito prima dell'indice del documento. La tabella deve registrare le seguenti informazioni:

- **Versione del file.**
- **Data di rilascio.**
- **Autore.**
- **Verificatore.**
- **Descrizione:** un riassunto delle modifiche apportate.

La convenzione per il versionamento è presente al paragrafo : 2.3.2 .

### 2.1.3.1 Verballi: struttura generale

Questo documento assume l'importante ruolo di costituire un registro ufficiale, atto a riportare in maniera accurata gli argomenti trattati, le decisioni adottate, le azioni da intraprendere e le figure coinvolte durante i meeting.

In particolare, è possibile distinguere tra verballi interni, relativi a meeting in cui sono presenti solo i componenti dell'organizzazione, e verballi esterni, che trovano applicazione quando vi sono terze parti coinvolte nelle discussioni o nelle decisioni documentate.

Nella prima pagina oltre alle informazioni comuni a ogni documento vengono specificate:

- Data della riunione e tipologia (Interna, Esterna) nel nome del documento;

- Luogo: canale di comunicazione adottato;
- Ora di inizio e fine dell'incontro;
- Amministratore;
- Partecipanti della riunione (interni ed esterni);
- Il Responsabile (in basso a destra);

Corpo del documento:

- **Revisione del periodo precedente:** Si valutano gli aspetti positivi, le difficoltà incontrate e si identificano azioni di miglioramento per ottimizzare i processi;
- **Ordine del giorno:** Elenco delle tematiche discusse durante la riunione, accompagnate dai relativi esiti.  
Nel caso del verbale esterno, se sono presenti richieste di chiarimenti effettuate alle terze parti coinvolte, saranno incluse nella sottosezione:
  - **Richieste e chiarimenti.**
- **Attività da svolgere:** Tabella dove viene specificato:
  - Nome della task da svolgere;
  - Id issue del ITS;
  - Verificatore dell'attività.

Ultima pagina:

- Nel caso di verbale esterno, in ultima pagina, deve essere presente la firma delle terze parti coinvolte, il luogo e la data.

Il template dei verbali è disponibile al link:

<https://github.com/ByteOps-swe/Sorgente-documenti/tree/main/Documents/Verbali/Templates>

## 2.1.4 Regole tipografiche

### Documenti del progetto e nome dei File

Il nome dei documenti generati deve essere omogeneo, con la prima lettera maiuscola ed il resto minuscolo e deve contenere un riferimento alla versione del documento (notazione di versionamento: 2.1.3).

Nello specifico devono seguire la seguente convenzione:

- **Verbali interni:** Verbale\_interno\_DD-MM-AAA ;
- **Verbali esterni:** Verbale\_esterno\_DD-MM-AAA ;
- **Norme di Progetto:** Norme\_di\_progetto\_vX.Y.Z ;
- **Analisi dei requisiti:** Analisi\_dei\_requisiti\_vX.Y.Z ;
- **Piano di progetto:** Piano\_di\_progetto\_vX.Y.Z ;
- **Glossario:** Glossario\_vX.Y.Z .

#### Regole sintattiche:

- I titoli delle sezioni iniziano con la lettera maiuscola;
- Viene inserito ';' alla fine delle voci dell'elenco tranne l'ultima che termina con '.' . Ogni voce dell'elenco inizia con una lettera maiuscola;
- Le date vengono scritte nel formato GG/MM/AAAA.
- I numeri razionali si scrivono utilizzando la virgola come separatore tra parte intera e parte decimale.

#### Stile del testo:

- **Grassetto:** Titoli delle sezioni, parole o frasi ritenute di rilievo.
- **Italico:** Riferimento a paragrafi o sezioni, nomi delle aziende e nomi propri dei membri del team.

### 2.1.5 Abbreviazioni

Nei documenti vi sono molte ripetizioni di termini per la quale si possono usare le seguenti abbreviazioni:

Abbreviazione	Scrittura Estesa
RTB	Requirements and Technology Baseline
PB	Product Baseline
CA	Customer Acceptance
ITS	Issue Tracking System
CI	Configuration Item
SAL	Stato Avanzamento Lavori

## 2.1.6 Strumenti

Gli strumenti utilizzati dalle attività di redazione dei documenti vogliono soddisfare il principio adottato di "Documentation as Code".

- **GitHub**: versionamento, collaborazione, integrazione continua, automazione e distribuzione;
- **Latex**: scrittura in formato testuale, linguaggio per la stesura di documenti compilati;
- **Overleaf**: per la redazione dei documenti in  $\text{\LaTeX}_G$  collaborativa;
- **VSCode**: per la redazione con l'utilizzo del plugin  $\text{\LaTeX}$ Workshop.

## 2.2 Verifica

### 2.2.1 Verifica dei documenti

#### 2.2.1.1 I verificatori

Il ruolo del verificatore nei documenti è cruciale per garantire la qualità e l'accuratezza del contenuto.

Quando il verificatore individua un'Issue nella colonna "Da revisionare" della **DashBoard documentazione**, sarà tenuto a convalidare il file corrispondente presente nella repository "Sorgente documenti".

In aggiunta, il revisore riceverà una notifica via email quando il redattore completa la propria attività, comunicandogli la presenza della pull request assegnatagli.

Nella sezione "pull request" di GitHub, il revisore troverà la richiesta di unire il branch di redazione al branch "main", assumendo il ruolo di revisore. Accedendo alla pull request su GitHub, il revisore avrà la possibilità di esaminare attentamente il documento in questione e di aggiungere commenti visibili ai redattori nel caso in cui siano necessarie modifiche per la validazione.

Per validare un documento i le verifiche da attuare sono:

- **Revisione della Correttezza Tecnica**: Revisione tecnica del documento per garantire che tutte le informazioni siano corrette, coerenti e rispettino le norme stabilite.
- **Conformità alle Norme**: Verifica che il documento segua le linee guida e gli standard prestabiliti per la formattazione, la struttura e lo stile.
- **Consistenza e Coerenza**: Si assicura che il documento sia consistente internamente e coerente con altri documenti correlati. Verifica che non ci siano discrepanze o contraddizioni.

- **Chiarezza e Comprensibilità:** Valuta la chiarezza del testo, assicurandosi che il linguaggio sia comprensibile per il pubblico di destinazione e che non ci siano ambiguità.
- **Revisione Grammaticale e Ortografica** Controlla la grammatica, l'ortografia e la punteggiatura del documento per garantire una presentazione professionale.

Dopo l'attuazione dei controlli sopra citati e verificato il loro rispetto, i passi per convalidare il documento sono i seguenti:

1. **Accetta la Pull Request:** Accedi alla pagina della pull request in cui agisci come revisore nel repository "Sorgente documenti" su GitHub » Risolvi eventuali conflitti » Merge Pull Request;
2. **Elimina il branch:** Elimina il branch creato per la redazione del documento (o sezione).
3. **Sposta la issue in Done:** Nella **DashBoard documentazione** sposta la issue relativa al documento validato da "Da revisionare" a "Done".
4. **Generazione PDF:** Un automazione tramite GitHub Actions compilerà il file  $\text{\LaTeX}$  e genererà automaticamente il PDF nel branch main della repository **Documents** con, se richiesto, la versione aggiornata nel nome.

## 2.3 Gestione della configurazione

### 2.3.1 Introduzione

La gestione della configurazione del progetto è una attività che norma il tracciamento e il controllo delle modifiche a documenti e prodotti del software detti Configuration Item (CI). La gestione della configurazione può essere applicata a qualunque categoria di documenti o di "artefatti" che svolga un ruolo nello sviluppo software. Secondo lo standard IEEE, la gestione della configurazione del software è un processo di identificazione, organizzazione e controllo delle modifiche apportate ai prodotti software durante il loro ciclo di vita. Lo standard IEEE 828-2012 definisce la gestione della configurazione del software come "un processo disciplinato per gestire l'evoluzione del software".

### 2.3.2 Numeri di versionamento

#### Documentazione:

La convenzione per identificare la versione di un documento è: X.Y.Z con :

- **X:** Viene incrementato al raggiungimento di RTB, PB ed eventualmente CA;



- **Y:** Viene incrementato quando vengono apportate modifiche significative al documento, come cambiamenti strutturali, nuove sezioni importanti o modifiche sostanziali nel contenuto;
- **Z:** Viene incrementato per modifiche minori o aggiornamenti al documento. Questi potrebbero includere correzioni di errori, miglioramenti marginali o l'aggiunta di nuovi contenuti meno rilevanti.

L'incremento dei valori più significativi porta i meno significativi a zero.

Ogni variazione di versione deve essere presente nel registro delle modifiche.

### 2.3.3 Tecnologie

Le tecnologie adottate per la gestione dei configuration item sono:

- **Git:** Version Control System distribuito utilizzato per il versionamento dei CI;
- **GitHub:** Piattaforma web che utilizza Git per il controllo di versione dei CI e per il Ticketing. Utilizzata per la gestione dei change request tramite issue e label e per la contabilità dello stato di configurazione (Branching, posizionamento nella repository e DashBoard di progetto).

### 2.3.4 Repository

Le repository del team sono:

- **Sorgente-documenti:** Repository per il versionamento, la gestione e lo sviluppo dei sorgenti della documentazione;
- **Documents:** Repository destinata ai commentanti/proponenti dove vengono condivisi solo i PDF dei file sorgenti revisionati.
- **proof-of-concept:** Repository destinata al POC.

#### 2.3.4.1 Struttura repository

I repository destinati alla documentazione sono organizzati come segue:

- **Candidatura:** Contenente i documenti richiesti per la candidatura;
  - **Verbali:** Contiene tutti i verbali redatti nel corso del periodo di candidatura, distinti tra verbali esterni e interni;
  - Lettera di presentazione;
  - Valutazione dei costi e assunzione impegni;

- Valutazione dei capitolati.
- **RTB:** Contenente i documenti richiesti per la revisione omonima;
  - **Verbali:** Contiene tutti i verbali redatti nel corso del periodo della RTB, distinti tra verbali esterni e interni;
  - Piano di Qualifica;
  - Piano di Progetto;
  - Analisi dei Requisiti;
  - Glossario;
  - Norme di Progetto.
- **PB.**

#### 2.3.4.2 Sincronizzazione e Branching

**Documentazione:** Il processo operativo utilizzato per la redazione della documentazione, noto come "feature branch workflow", implica la creazione di un ramo dedicato per ciascun documento o sezione da elaborare. Tale metodologia permette una parallelizzazione agile dei lavori evitando sovrascritture indesiderate di altri lavori e permette l'adozione dei principi "Documentation as code".

### 2.3.5 Controllo di configurazione

#### 2.3.5.1 Change request (Richiesta di modifica)

Seguendo lo standard ISO/IEC 12207 per affrontare questo processo in modo strutturato le attività sono:

1. **Identificazione e registrazione:** Le change request vengono identificate, registrate e documentate accuratamente. Questo include informazioni come la natura della modifica richiesta, l'urgenza, l'impatto sul sistema esistente  
L'identificazione avviene tramite la creazione di un issue nell'ITS con label: "Change request";
2. **Valutazione e analisi:** Le change request vengono valutate dal team per determinare la loro fattibilità, importanza e impatto sul progetto. Si analizzano i costi e i benefici associati all'implementazione della modifica;
3. **Approvazione o rifiuto:** Il responsabile valuta le informazioni raccolte e decide se approvare o respingere la change request. Questa decisione può essere basata su criteri come il budget, il tempo, le priorità degli stakeholder;

4. **Pianificazione delle modifiche:** Se una change request viene approvata, viene pianificata e integrata nel ciclo di sviluppo del software. Questo può richiedere la rinegoziazione dei tempi di consegna, la revisione del piano di progetto, ecc;
5. **Implementazione delle modifiche:** Le modifiche vengono effettivamente implementate. Durante questo processo, è fondamentale mantenere una traccia accurata di ciò che viene fatto per consentire una corretta documentazione e, se necessario, la possibilità di un rollback;
6. **Verifica e validazione:** Le modifiche apportate vengono verificate per assicurarsi che abbiano raggiunto gli obiettivi previsti e non abbiano introdotto nuovi problemi o errori;
7. **Documentazione:** Tutti i passaggi del processo di gestione delle change request vengono documentati accuratamente per garantire la trasparenza e la tracciabilità. Questa documentazione è utile per futuri riferimenti e per l'apprendimento dalle modifiche apportate;
8. **Comunicazione agli interessati:** Durante tutto il processo, è importante comunicare in modo chiaro e tempestivo agli interessati, come il team di sviluppo, i clienti e altri stakeholder, per mantenere la trasparenza e la fiducia.

### 2.3.6 Configuration Status Accounting (Contabilità dello Stato di Configurazione)

Questo processo si occupa di registrare e tenere traccia dello stato di tutte le configurazioni di un sistema software durante il suo ciclo di vita.

- **Registrazione delle configurazioni:** Registrazione delle informazioni dettagliate su ogni elemento di configurazione;
  - **Documentazione:** Le informazioni del documento sono presenti nella prima pagina di ciascuno.
- **Stato e cambiamenti:** Tenere traccia dello stato attuale di ciascun elemento di configurazione e di tutti i cambiamenti che avvengono nel corso del tempo. Ciò include le versioni attuali, le revisioni, le modifiche e le baselines;
  - **Registro delle modifiche:** Per monitorare lo stato di ciascun CI, si utilizza il registro delle modifiche incorporato in ognuno di essi;
  - **Branching & Dashboard:** Per visionare la presenza di lavori in atto su di un CI si utilizza a supporto la presenza di branch attivi che lo riguardano e il posizionamento di issue associate nella Dashboard di progetto.

\* Ogni issue è associato ad un CI tramite label.

- **Supporto per la gestione delle change request:** Registra e documenta le modifiche apportate agli elementi di configurazione in risposta alle richieste di modifica.
  - **ITS:** Per supportare i change request viene utilizzato l'ITS di github, l'identificazione avviene tramite la creazione di un issue nell'ITS con label: "Change request".

La Contabilità dello Stato di Configurazione è un processo fondamentale per mantenere la trasparenza e la tracciabilità nel ciclo di vita del software, aiutando a gestire le configurazioni in modo coerente e a mantenere un registro accurato di tutte le attività e le modifiche che coinvolgono gli elementi di configurazione.

Ad esempio, l'aggiunta di un elemento di configurazione al ramo principale (main) lo designa come la versione più recente e revisionata, mentre le versioni (X.0.0) con  $X \geq 1$  indicano l'appartenenza del suddetto elemento di configurazione alla baseline, con la medesima versione.

### 2.3.7 Release management and delivery

**Documentazione:** Dopo aver portato a termine le attività nel proprio branch, il responsabile del suo sviluppo è tenuto a avviare una richiesta di pull per incorporare le modifiche nel ramo principale. La richiesta di pull può essere accettata solo se la verifica ha esito positivo.

## 3 Processi organizzativi

Questa sezione fornisce linee guida e normative per la pianificazione, l'esecuzione e il monitoraggio dei processi, contribuendo così a garantire la coerenza e la qualità nelle attività di sviluppo del software.

### 3.1 Pianificazione

In questa sezione saranno delineati i procedimenti che conducono dalla definizione dei ruoli alla precisa assegnazione delle attività ai singoli membri del gruppo.

#### 3.1.1 Assegnazione dei ruoli

Durante l'intero periodo del progetto, i membri del gruppo assumeranno sei ruoli distinti, ovvero assumeranno le responsabilità e svolgeranno le mansioni tipiche dei professionisti nel campo dello sviluppo software.

I ruoli a disposizione sono:

### 3.1.1.1 Responsabile

Figura fondamentale che coordina il gruppo, fungendo da punto di riferimento per il committente e il fornitore e svolgendo il ruolo di mediatore tra le due parti.

In particolare si occupa di:

- Gestire le relazioni con l'esterno;
- Pianificare le attività: quali svolgere, data di inizio e fine, assegnazione delle priorità...
- Valutare i rischi delle scelte da effettuare;
- Controllare i progressi del progetto;
- Gestire le risorse umane;
- Approvazione della documentazione;

### 3.1.1.2 Amministratore

Questa figura professionale è incaricata del controllo e dell'amministrazione dell'ambiente di lavoro utilizzato dal gruppo ed è anche il punto di riferimento per quanto concerne le norme di progetto. Le sue mansioni principali sono:

- Affrontare e risolvere le problematiche associate alla gestione dei processi;
- Gestire versionamento della documentazione;
- Gestire la configurazione del prodotto;
- Redigere ed attuare le norme e le procedure per la gestione della qualità;
- Amministrare le infrastrutture e i servizi per i processi di supporto;

### 3.1.1.3 Analista

Figura professionale con competenze avanzate riguardo l'attività di analisi dei requisiti ed il dominio applicativo del problema. Il suo ruolo cruciale è quello di identificare, documentare e comprendere a fondo le esigenze e le specifiche del progetto, traducendole in requisiti chiari e dettagliati. Si occupa di:

- Analizzare il contesto di riferimento, definire il problema in esame e stabilire gli obiettivi da raggiungere;
- Comprendere il problema e definire la complessità e i requisiti;
- Redigere il documento *Analisi dei Requisiti*;
- Studiare i bisogni espliciti ed impliciti;

#### 3.1.1.4 Progettista

Il *Progettista* è la figura di riferimento per quanto riguarda le scelte progettuali partendo dal lavoro dell'analista. Spetta al progettista assumere decisioni di natura tecnica e tecnologica, oltre a supervisionare il processo di sviluppo. Tuttavia, non è responsabile della manutenzione del prodotto. In particolare si occupa di:

- Progettare l'architettura del prodotto secondo specifiche tecniche dettagliate;
- Prendere decisioni per sviluppare soluzioni che soddisfino i criteri di affidabilità, efficienza, sostenibilità e conformità ai Requisiti;
- Redige la *Specifica Architetture* e la parte pragmatica del *Piano di Qualifica*;

#### 3.1.1.5 Programmatore

Il programmatore è la figura professionale responsabile della codifica del software. Il suo ruolo principale consiste nell'implementare codice informatico basato sulle specifiche fornite dall'analista e sull'architettura fornita dal progettista. In particolare, il *Programmatore*:

- Scrivere codice informatico mantenibile in conformità con le *Specifiche di Progetto*;
- Codificare le varie parti dell'architettura elaborata seguendo l'architettura ideata dal Progettista;
- Realizza gli strumenti per verificare e validare il codice;
- Redigere il *Manuale Utente*;

#### 3.1.1.6 Verificatore

La principale responsabilità del *Verificatore* consiste nell'ispezionare il lavoro svolto da altri membri del team per assicurare la qualità e la conformità alle attese prefissate. Stabilisce se il lavoro è stato svolto correttamente sulla base delle proprie competenze tecniche, esperienza e conoscenza delle norme. Si occupa di:

- Verificare che i prodotti siano conformi alle *Norme di Progetto*;
- Verificare la conformità dei prodotti ai requisiti funzionali e di qualità;
- Ricercare ed in caso segnalare eventuali errori;
- Redigere la sezione retrospettiva del *Piano di Qualifica*, descrivendo le verifiche e le prove effettuate durante il processo di sviluppo del prodotto;

### 3.1.1.7 Ticketing

2.3.6 GitHub è adottato come sistema di tracciamento degli issue (ITS), garantendo così una gestione agevole e trasparente dei compiti da svolgere. L'amministratore ha la facoltà di creare e assegnare specifici compiti identificati dal responsabile, assicurando chiarezza sulle responsabilità di ciascun individuo e stabilendo tempi definiti. Inoltre, ogni membro del gruppo può monitorare i progressi compiuti nel periodo corrente, consultando lo stato di avanzamento dei vari task attraverso le Dashboard:

- **DashBoard:** Per una chiara visione dello stato dei task;
- **RoadMap:** Per visione delle scadenze dei task.

La procedura da seguire in caso di necessità di svolgere un compito è la seguente:

1. L'amministratore crea e assegna il task su GitHub, seguendo la convenzione descritta in seguito;
2. L'incaricato apre una branch su GitHub seguendo la denominazione suggerita;
3. All'inizio del lavoro di produzione il task viene marcato in corso sulla Dashboard GitHub;
4. Finito il lavoro di produzione viene aperta la pull request su GitHub assegnando il verificatore.
5. Il task viene marcato nella colonna "Da revisionare" sulla Dashboard GitHub.
6. Il verificatore si accerta e agisce secondo quanto esposto al punto 2.2 nel caso di appartenenza;
7. Il task viene marcato nella colonna "Done" della Dashboard GitHub.

I task sono creati dall'amministratore e hanno i seguenti attributi:

- **Titolo:** un titolo conciso e descrittivo;
  - **Descrizione:**
    - Descrizione testuale di "to-do";
    - Come ultima riga il Verificatore della task. (Verificatore: Mario Rossi.)
  - **Assegnatario:** incaricato svolgimento della task;
  - **Scadenza:** Data entro la quale la task deve essere completata;
  - **Labels:** Tag per identificare la categoria della task. (ex. Verbale, Documents, Develop, Bug).
- Inoltre per associare ad ogni Issue un CI vengono utilizzati i seguenti label:

- **NdP:** Norme di progetto;
  - **PdQ:** Piano di Qualifica;
  - **PdP:** Piano di Progetto;
  - **AdR:** Analisi dei Requisiti;
  - **Poc:** Proof of concept;
  - **Gls:** Glossario.
- **Milestone:** Milestone associata alla task.

## 3.2 Comunicazione

Il gruppo *ByteOps* mantiene comunicazioni attive, sia interne che esterne al team, le quali possono essere sincrone o asincrone a seconda delle necessità.

### 3.2.1 Comunicazioni sincrone

Per le comunicazioni interne ed esterne, il gruppo *ByteOps*, in collaborazione con l'azienda proponente, ha scelto di adottare *Discord<sub>G</sub>* in quanto permette di comunicare tramite chiamate vocali, videochiamate, messaggi di testo, media e file in chat private o come membri di un "server *Discord*".

### 3.2.2 Comunicazioni asincrone

#### 3.2.2.1 comunicazioni asincrone interne

Le comunicazioni asincrone interne al nostro team avvengono tramite l'applicazione *Telegram<sub>G</sub>* all'interno di un Gruppo dedicato, il quale consente una comunicazione rapida tra tutti i membri del gruppo. Inoltre, tramite la stessa piattaforma, è possibile avere conversazioni dirette e private (*chat*) tra due membri.

#### 3.2.2.2 comunicazioni asincrone esterne

Per le comunicazioni asincrone esterne sono stati adottati due canali differenti:

- **E-mail** La posta elettronica è stata utilizzata per comunicare con il committente e, nelle prime fasi del progetto, per coordinare gli incontri con l'azienda proponente. Questa forma di comunicazione offre la flessibilità necessaria per coordinare incontri sincroni e revisioni in modo efficace, consentendo a entrambe le parti di pianificare le interazioni in base alla propria disponibilità.



- **Element** E' un client di messaggistica istantanea libero ed open source che supporta conversazioni strutturate e crittografate. La sua flessibilità nell'adattarsi a varie esigenze di comunicazione, inclusa la possibilità di condividere file, immagini e altri documenti, ha reso la piattaforma un'opzione versatile e completa per soddisfare le esigenze specifiche del nostro contesto lavorativo.

## 3.3 Gestione degli incontri

### 3.3.1 Incontri formali esterni

Durante l'implementazione del progetto, si renderà necessaria l'organizzazione di vari incontri con i *Committenti* e/o il *Proponente* allo scopo di valutare lo stato di avanzamento del prodotto.

La convocazione di tali incontri è di competenza del *Responsabile di Progetto*, il quale è incaricato di pianificarli e di agevolarne lo svolgimento in maniera efficiente. Sarà compito del Responsabile anche l'esposizione dei punti di discussione al Proponente, lasciando la parola ai membri del gruppo interessati quando necessario. In aggiunta, sarà designato un *Segretario* con l'incarico di redigere una sintesi dei temi discussi.

Questo approccio assicura una comunicazione efficace tra il nostro team e i rappresentanti aziendali, garantendo una gestione ottimale del tempo e una registrazione accurata delle informazioni rilevanti emerse durante gli incontri.

### 3.3.2 Incontri formali interni

#### 3.3.2.1 Meeting settimanali interni

Il nostro team ha adottato un approccio settimanale attraverso incontri simili a "stand-up meetings" al fine di facilitare una comunicazione costante e coordinare il progresso delle attività interne.

Questi incontri rivestono un ruolo cruciale nel monitorare il progresso delle mansioni assegnate, valutare i risultati conseguiti e affrontare le sfide che possono sorgere. Durante questi momenti, i membri del team condividono gli aggiornamenti sulle proprie attività, identificano le problematiche riscontrate e discutono di opportunità di miglioramento nei processi di lavoro. Questo ambiente aperto e collaborativo favorisce l'innovazione e la condivisione di nuove prospettive.

Per agevolare la comunicazione sincrona, il canale utilizzato per questi incontri è *Discord*, ritenuto particolarmente efficace per tali scopi.