ZPUs

ENGINEERING TEAM

7zpus.swe@gmail.com

Norme di Progetto

Indice

1 Introduzione **7**
- 1.1 Scopo . . . 7
- 1.2 Glossario . . . 7
- 1.3 Riferimenti . . . 7
  - 1.3.1 Riferimenti Normativi . . . 7
  - 1.3.2 Riferimenti Informativi . . . 7

2 Processi Primari **8**
- 2.1 Processo di Fornitura . . . 8
  - 2.1.1 Attività di processo . . . 8
    - 2.1.1.1 Avvio . . . 8
    - 2.1.1.2 Preparazione della proposta di fornitura . . . 8
    - 2.1.1.3 Accordo . . . 8
    - 2.1.1.4 Pianificazione . . . 9
    - 2.1.1.5 Esecuzione e controllo . . . 9
    - 2.1.1.6 Verifica e validazione . . . 9
    - 2.1.1.7 Consegna e terminazione . . . 9
    - 2.1.1.8 Accordi con l'azienda proponente . . . 9
  - 2.1.2 Documentazione fornita . . . 10
    - 2.1.2.1 Analisi dei capitolati . . . 10
    - 2.1.2.2 Lettera di presentazione . . . 10
    - 2.1.2.3 Verbali interni ed esterni . . . 10
    - 2.1.2.4 Lettera di candidatura . . . 10
    - 2.1.2.5 Analisi dei requisiti . . . 10
    - 2.1.2.6 Glossario . . . 11
    - 2.1.2.7 Piano di progetto . . . 11
    - 2.1.2.8 Piano di qualifica . . . 11
    - 2.1.2.9 Lettera di presentazione . . . 12
  - 2.1.3 Strumenti . . . 12

2.2 Processo di sviluppo 12
2.2.1 Attività di processo 12
2.2.2 Analisi dei Requisiti 13
2.2.3 Casi d'uso 13
2.2.4 Requisiti 14
2.3 Processo operativo 14
2.3.1 Pianificazione operativa 14
2.3.2 Gestione dei rischi operativi 15
2.4 Processo di manutenzione 15
2.4.1 Manutenzione correttiva 16
2.4.2 Manutenzione adattiva 16
2.4.3 Manutenzione preventiva 16
2.4.4 Identificazione della necessità di manutenzione 16
2.4.5 Verifica e validazione 16
2.4.6 Metriche 17

3 Processi di Supporto 17
3.1 Processo di documentazione 17
3.1.1 Attività: Pianificazione della documentazione 17
3.1.1.1 Procedure di Pianificazione 17
3.1.1.2 Strumenti di Pianificazione 18
3.1.2 Attività: Produzione della documentazione 18
3.1.2.1 Linee guida per la produzione della documentazione . . 18
3.1.2.1.1 Denominazione e datazione documentazione . 19
3.1.2.2 Strumenti di Produzione 19
3.1.3 Attività: Revisione e Approvazione 19
3.1.3.1 Procedure di Revisione e Approvazione 20
3.1.3.2 Strumenti di Revisione e Approvazione 21
3.2 Processo di Gestione della Configurazione 21
3.2.1 Attività: Identificazione della Configurazione 21
3.2.1.1 Procedure: Identificazione degli SCIs 21
3.2.1.1.1 Issue e SubIssue 22
3.2.1.2 Strumenti di Identificazione 22
3.2.2 Attività: Controllo della configurazione 23
3.2.2.1 Procedure: Smart Commit 25
3.2.2.2 Strumenti di Controllo delle Configurazioni 26
3.2.3 Attività: Versionamento e Identificazione 26
3.2.3.1 Procedure: Standard per le Branch 26
3.2.3.2 Strumenti di Versionamento 26
3.2.4 Registrazione delle Configurazioni 26
3.2.4.1 Procedure: Registrazione delle Modifiche 26
3.3 Processo di verifica 27
3.3.1 Verifica documentale 27
3.3.2 Analisi Statica 28

3.3.3 Analisi Dinamica 28
3.4 Processo di validazione 29
3.5 Processo di revisione congiunta 29
3.5.1 Attività: Pianificazione delle revisioni 29
3.5.1.1 Procedure di Revisione 29
3.5.2 Attività: Preparazione dei materiali 29
3.5.2.1 Procedure di preparazione 29
3.5.3 Attività: Conduzione della revisione 30
3.5.3.1 Procedure di conduzione 30
3.5.4 Attività: Post incontro 30
3.5.4.1 Procedure post incontro 30
3.5.5 Strumenti di revisione congiunta 30
3.6 Processo di risoluzione dei problemi 31
3.6.1 Attività: Identificazione del problema 31
3.6.1.1 Procedure di identificazione 31
3.6.2 Attività: Analisi e classificazione 32
3.6.2.1 Procedure di analisi e classificazione 32
3.6.3 Attività: Sviluppo della soluzione 32
3.6.3.1 Procedure di sviluppo della soluzione 32
3.6.4 Attività: Implementazione e verifica 32
3.6.4.1 Procedure di implementazione e verifica 32
3.6.5 Attività: Procedure generali di risoluzione 33
3.6.5.1 Procedure generali di risoluzione 33
3.6.6 Integrazione con altri processi 33
3.6.7 Strumenti di risoluzione dei problemi 33
3.7 Gestione della qualità 34
3.7.1 Attività: Pianificazione della qualità 34
3.7.1.1 Procedure di pianificazione 34
3.7.2 Attività: Monitoraggio continuo 34
3.7.2.1 Procedure di monitoraggio 34
3.7.3 Attività: Valutazione periodica 35
3.7.3.1 Procedure di valutazione 35
3.7.4 Attività: Azioni correttive 36
3.7.4.1 Procedure per azioni correttive 36
3.7.5 Attività: Procedure generali di gestione della qualità 36
3.7.5.1 Procedure di gestione della qualità 36
3.7.6 Metriche di gestione della qualità 37
3.7.7 Strumenti di gestione della qualità 37

4 Processi Organizzativi **37**
4.1 Gestione 37
4.1.1 Ruoli di progetto 37
4.1.2 Attività 38
4.2 Gestione 38

4.3 Infrastruttura 38
4.3.1 Attività di processo 38
4.3.2 Procedure di processo 39
4.3.2.1 Strumenti di Creazione 39
4.3.2.2 Jira 39
4.3.2.3 GitHub 39
4.3.3 Attività: Manutenzione 39
4.3.3.1 Procedure di Manutenzione 39
4.3.3.2 Strumenti di Manutenzione 40
4.3.3.3 Criteri di Scelta degli Strumenti 40
4.4 Knowledge Base 41
4.4.0.1 GitHub 41
4.4.0.2 Jira 41
4.4.0.2.1 Automation 41
4.4.0.3 VSCode 42
4.4.0.3.1 Atlassian: Jira, Rovo Dev, Bitbucket 42
4.4.0.3.2 GitHub Pull Requests and Issues 42
4.4.0.4 Google Presentazioni 42
4.4.0.5 Google Drive 43
4.4.0.6 Google Mail 43
4.4.0.7 Discord 43
4.4.0.8 Whatsapp 43
4.5 Processo di Miglioramento 43
4.5.1 Attività: Analisi degli Incidenti 43
4.5.1.1 Procedure di Analisi degli Incidenti 44
4.5.2 Attività: Monitoraggio degli Strumenti 44
4.5.2.1 Procedure di Monitoraggio 44
4.5.2.2 Strumenti di Monitoraggio 44
4.5.3 Attività: Modifiche all'Infrastruttura 45
4.5.3.1 Procedure di Modifica 45
4.5.3.2 Strumenti di Modifica 45
4.5.4 Frequenza dei Cicli di Miglioramento 45
4.5.5 Metriche di Miglioramento 46
4.6 Processo di Formazione 46
4.6.1 Attività: Pianificazione della Formazione 46
4.6.1.1 Procedure di Pianificazione 46
4.6.2 Attività: Erogazione della Formazione 46
4.6.2.1 Procedure di Erogazione 47
4.6.3 Attività: Valutazione della Formazione 47
4.6.3.1 Procedure di Valutazione 47
4.6.3.2 Strumenti di Valutazione 47

5 Metriche della qualità 48
5.1 Qualità di Processo 48

5.1.1 Processi primari 49
5.1.1.1 Processo di fornitura 49
5.1.1.2 Processo di Sviluppo 50
5.1.1.3 Processo di Integrazione 51
5.1.1.4 Processo di Documentazione 51
5.1.1.5 Processo di Verifica 52
5.1.2 Processi Organizzativi 52
5.1.2.1 Processo di Gestione dei Rischi 52
5.1.2.2 Processo di Gestione della qualità 52
5.2 Qualità di Prodotto 53
5.2.0.1 Funzionalità 53
5.2.0.2 Affidabilità 54
5.2.0.3 Usabilità 55
5.2.0.4 Efficienza 55
5.2.0.5 Manutenibilità 57
5.2.0.6 Portabilità 58
5.2.0.7 Portabilità 58
5.3 Strumenti 58
5.3.0.1 Jira Software 58
5.3.0.2 GitHub & GitHub Actions 58
5.3.1 Strumenti di Analisi Statica e Qualità del Codice 59
5.3.1.1 SonarQube 59
5.3.1.2 Aspell & Script Python (Custom) 59
5.3.2 Strumenti di Testing e Performance (Electron-Angular) 59
5.3.2.1 Cypress 59
5.3.2.2 Electron Manager & Chrome DevTools Protocol 59
5.3.2.3 TotalValidator 60
5.3.2.4 Funzioni interne di misurazione performance 60

Elenco delle figure

1 Commit della PR con Smart Commit verso le due issue 20
2 Flusso del processo di controllo 23
3 Creazione del branch di lavoro tramite estensione Jira in VSCode 24
4 Creazione della PR verso l'issue branch *in_lavorazione* 25
5 Modello a V 48

Elenco delle tabelle

Tabella di Versionamento

Versione	Data	Autore	Verificatore	Descrizione
0.14	2026/02/11	Vigolo Davide	Soligo Lorenzo	Definizione metriche di qualità
0.13	2026/02/09	Fattoni Antonio	Vigolo Davide	Correzione processo di documentazione e controllo della configurazione
0.12	2026/02/08	Vigolo Davide	Soligo Lorenzo	Definizione metriche di qualità
0.11	2026/02/06	Laoud Zakaria	Soligo Lorenzo	Stesura paragrafi 3.6, 3.7, 3.8
0.10	2025/01/25	Aaron Gingillino	Georgescu Diana	Scrittura paragrafi 2.2, 4.1
0.9	2025/01/25	Aaron Gingillino	Fattoni Antonio	Scrittura paragrafi 3.3, 3.4
0.8.1	2026/01/19	Soligo Lorenzo	Laoud Zakaria	Fine scrittura paragrafi 4.2, 4.3, 4.4
0.8	2026/01/11	Soligo Lorenzo	Laoud Zakaria	Stesura paragrafi 4.2, 4.3, 4.4
0.7	2025/12/15	Rocco Matteo A.	Georgescu Diana	Stesura paragrafo 5
0.6	2025/12/10	Georgescu Diana	Soligo Lorenzo	Stesura sottosezioni 2.3, 2.4
0.5	2025/12/01	Soligo Lorenzo	Rocco Matteo A.	Aggiornamento nuovo standard per gestione branch, convenzioni sui nomi, date e versioni. Sezioni 3.1.2, 3.1.3, 3.2.1.1.1, 3.2.4, 4.2.3.2.1
0.4.1	2025/11/28	Soligo Lorenzo	Fattoni Antonio	Correzione nella Procedura di Revisione Paragrafo 3.1.3.1 e aggiornamento immagini
0.4	2025/11/26	Soligo Lorenzo	Fattoni Antonio	Ristrutturazione completa Processi (ISO 12207: Attività-Procedure-Strumenti)
0.3	2025/11/25	Soligo Lorenzo	Fattoni Antonio	Creazione e stesura sezioni Processi di Infrastruttura e sottosezioni 4.2.1-4.3. Nuova struttura Paragrafi e sottoParagrafi
0.2	2025/11/22	Soligo Lorenzo	Fattoni Antonio	Creazione e stesura sezioni Documentazione e sottosezioni 3.1-3.1.5
0.1	2025/11/16	Rocco Matteo A.	Soligo Lorenzo	Creazione e stesura sezioni Introduzione e Processo di fornitura

1 Introduzione

1.1 Scopo

Questo documento ha l'obiettivo di definire e normare il *Way of Working*$_G$, ovvero le regole di lavoro che ogni membro del gruppo deve rispettare durante lo svolgimento delle *attività di progetto*$_G$ volte allo sviluppo dell'applicativo software ***DIPReader***$_G$, proposto dall'azienda *Sanmarco Informatica*$_G$. A ciascun membro è richiesto di seguirle integralmente per poter lavorare in maniera efficace, efficiente e omogenea. Data la natura incrementale della redazione del documento, il *responsabile di progetto*$_G$ ha il compito di mantenere aggiornate le presenti norme e gli eventuali riferimenti ad altri documenti in esse contenuti.

1.2 Glossario

Ogni termine tecnico o con un significato particolare nell'ambito dell'*Ingegneria del Software*$_G$, utilizzato nella documentazione di progetto, è definito nell'apposito documento Glossario 1.0 (ultimo accesso: 17/11/2025).

1.3 Riferimenti

Il gruppo ha redatto il presente documento in conformità con lo *standard*$_G$ ISO/IEC 12207:1995, integrandolo occasionalmente con approfondimenti tratti dalla sua versione più recente, ISO/IEC/IEEE 12207:2017, per includere dettagli aggiuntivi sugli approcci *agili*$_G$ e *iterativi*$_G$ che caratterizzano lo *sviluppo software*$_G$ moderno.

1.3.1 Riferimenti Normativi

- Standard ISO/IEC 12207:1995 (ultimo accesso: 17/11/2025)
- Standard ISO/IEC/IEEE 12207:2017
- Standard ISO/IEC/IEEE 24765:2017
- Capitolato C3: DIPReader (ultimo accesso: 13/11/2025)
- Regolamento di Progetto Didattico a.a. 2025/2026 (ultimo accesso: 17/11/2025)

1.3.2 Riferimenti Informativi

- Dispense del corso di Ingegneria del Software 2025/2026:
  - https://www.math.unipd.it/ tullio/IS-1/2025/Dispense/T01.pdf (ultimo accesso: 17/11/2025)
  - https://www.math.unipd.it/ tullio/IS-1/2025/Dispense/T02.pdf (ultimo accesso: 17/11/2025)
  - https://www.math.unipd.it/ tullio/IS-1/2025/Dispense/T03.pdf (ultimo accesso: 17/11/2025)
  - https://www.math.unipd.it/ tullio/IS-1/2025/Dispense/T04.pdf (ultimo accesso: 17/11/2025)

  - https://www.math.unipd.it/ tullio/IS-1/2025/Dispense/T05.pdf (ultimo accesso: 17/11/2025)
  - https://www.math.unipd.it/ tullio/IS-1/2025/Dispense/T06.pdf (ultimo accesso: 17/11/2025)
  - https://www.math.unipd.it/ tullio/IS-1/2025/Dispense/T07.pdf (ultimo accesso: 17/11/2025)
  - https://www.math.unipd.it/ tullio/IS-1/2025/Dispense/T08.pdf (ultimo accesso: 17/11/2025)
  - https://www.math.unipd.it/ tullio/IS-1/2025/Dispense/T09.pdf (ultimo accesso: 17/11/2025)
  - https://www.math.unipd.it/ tullio/IS-1/2025/Dispense/T10.pdf (ultimo accesso: 17/11/2025)
  - https://www.math.unipd.it/ tullio/IS-1/2025/Dispense/T11.pdf (ultimo accesso: 17/11/2025)
- Linee Guida Sviluppo Sicuro AGID (Agenzia per l'Italia Digitale)
- Linee Guida sulla formazione, gestione e conservazione dei documenti informatici AGID
- Documentazione LaTeX by Lorenzo Pantieri (ultimo accesso: 17/11/2025)
- Documentazione Jira

2 Processi Primari

2.1 Processo di Fornitura

Il *processo*$_G$ di fornitura contiene le attività e i compiti svolti dal *fornitore*$_G$. Per implementare correttamente il processo il gruppo si impegna a svolgere le seguenti attività.

2.1.1 Attività di processo

2.1.1.1 Avvio

Il fornitore analizza i *requisiti*$_G$ necessari alla proposta di fornitura, tenendo in considerazione eventuali vincoli organizzativi e normativi.

2.1.1.2 Preparazione della proposta di fornitura

Il fornitore prepara la proposta di fornitura in risposta alle richieste del committente e definisce i termini in cui si articola la proposta.

2.1.1.3 Accordo

Proponente e fornitore entrano nella fase di definizione dell'accordo di fornitura del prodotto software, prevedendo possibilità di negoziazione della fornitura da parte del fornitore.

2.1.1.4 Pianificazione

Il fornitore rielabora l'analisi dei requisiti fondamentali per definire il *framework*$_G$ entro il quale il prodotto verrà sviluppato e gestito, in modo tale da garantire un processo di qualità durante lo sviluppo. Si impegna inoltre a definire il modello del ciclo di vita del prodotto adatto alla complessità del progetto e ai relativi rischi che potrebbero insorgere. Tutte queste decisioni convergono nel Piano di Progetto.

2.1.1.5 Esecuzione e controllo

Il fornitore si impegna a sviluppare il prodotto secondo il Piano di Progetto, avendo cura di controllare che i processi siano stati eseguiti correttamente.

2.1.1.6 Verifica e validazione

Il fornitore stabilisce con la proponente le modalità di rendicontazione dello stato di avanzamento del prodotto e rende disponibili i documenti che dimostrino la verifica e validazione dei processi secondo i requisiti precedentemente individuati.

2.1.1.7 Consegna e terminazione

Il fornitore consegna il prodotto finale al proponente e ne espone le funzionalità.

2.1.1.8 Accordi con l'azienda proponente

I capitolati presentati dalle proponenti vengono analizzati e viene redatto il documento di Analisi dei capitolati, nel quale sono delineati i bisogni e i principali vincoli a cui attenersi per la fornitura del prodotto finale. Il fornitore espone ai committenti di fornitura, ovvero i Professori Vardanega Tullio e Cardin Riccardo, la Lettera di Presentazione della proposta di fornitura che descrive il preventivo di costi, cronogramma di sviluppo, suddivisione del lavoro e i ruoli coinvolti. La *proponente*$_G$, in qualità di *stakeholder*$_G$, esercita il diritto di ricevere la rendicontazione professionale e approfondita del lavoro svolto dal gruppo fornitore, perciò si instaura un accordo per delineare le modalità di comunicazione e il contenuto di tale rendicontazione. È previsto l'aggiornamento costante e tempestivo della proponente per quanto riguarda la pianificazione degli obiettivi e delle tempistiche di sviluppo individuate dal fornitore. Ogni qualvolta vi siano modifiche di notevole interesse esterno dal gruppo fornitore verranno comunicate all'azienda proponente attraverso appositi canali di comunicazione sincrona o asincrona. Il fornitore e la proponente hanno accordato lo svolgimento di un incontro di verifica dello stato di avanzamento lavori (*SAL*$_G$) in modalità sincrona ogni due settimane, in cui discutere l'andamento del lavoro e chiarire eventuali dubbi da parte del fornitore o segnalazioni di difformità dai requisiti iniziali della proponente. È inoltre sempre disponibile la comunicazione via email per questioni minori e di facile risoluzione. La consegna del prodotto è suddivisa in due *milestone*$_G$ principali: *RTB*$_G$ (Requirements and Technology Baseline) e *PB*$_G$ (Product Baseline).

2.1.2 Documentazione fornita

2.1.2.1 Analisi dei capitolati

Il documento di Analisi dei capitolati (ultimo accesso:) mette in evidenza le considerazioni fatte riguardo ai capitolati presentati. Vengono analizzati complessità, rischi e opportunità formative di ciascun capitolato per determinare la scelta ottimale per il gruppo.

2.1.2.2 Lettera di presentazione

Il documento di Lettera di presentazione (ultimo accesso:) formalizza la candidatura del team ai committenti per lo sviluppo del progetto proposto, dichiarando la data di consegna, il budget stimato e la composizione ufficiale del team.

2.1.2.3 Verbali interni ed esterni

I verbali interni ed esterni sono documenti che raccolgono le informazioni e le decisioni prese durante le riunioni di progetto, sia interne al team che con l'azienda proponente.

2.1.2.4 Lettera di candidatura

La lettera di candidatura è il documento che contiene la proposta formale di candidatura del team alla revisione RTB dove vengono esposti gli artefatti presenti nella baseline.

2.1.2.5 Analisi dei requisiti

Nel documento di Analisi dei requisiti (ultimo accesso: 17/11/2025) sono riportati i bisogni e i vincoli a cui attenersi per la realizzazione del prodotto finale. L'obiettivo è definire in maniera non ambigua i *casi d'uso*$_G$ (*Use Cases*) e i requisiti (*Requirements*) del software. Il documento è diviso nelle seguenti sezioni:

1. **Introduzione:** Include lo scopo, i riferimenti normativi (come il Capitolato C3) e informativi, e il rimando a un Glossario esterno per la terminologia tecnica.
2. **Descrizione:** Fornisce una panoramica del sistema, le sue funzionalità generali e l'identificazione degli utenti di destinazione.
3. **User Stories:** Elenca i bisogni degli utenti espressi in linguaggio naturale.
4. **Definizione dei casi d'uso:** Rappresenta il nucleo del documento, con i casi d'uso principali descritti tramite diagrammi UML e schede tecniche che includono attori, precondizioni, postcondizioni e flussi principali.
5. **Definizione dei requisiti:** Classificazione dettagliata dei requisiti individuati

2.1.2.6 Glossario

Il Glossario è il documento che raccoglie ogni termine di carattere tecnico, nomenclature e acronimi con particolare significato nell'ambito dell'Ingegneria del Software utilizzato nella documentazione di progetto. La definizione dei termini di glossario è coadiuvata dal contenuto dello standard ISO/IEC/IEEE 24765/2017.

2.1.2.7 Piano di progetto

Il Piano di progetto v1.0 (ultimo accesso: 17/11/2025) è il documento che espone all'esterno il lavoro di sviluppo svolto seguendo le procedure delineate all'interno di questo documento. Fornisce una guida dettagliata alla pianificazione, esecuzione e consuntivo delle attività completate in ciascuna *sprint*$_G$. Il documento è diviso nelle seguenti sezioni:

1. Introduzione
2. Analisi dei rischi e mitigazione
3. Modello di sviluppo
4. Pianificazione dei costi e suddivisione ruoli
5. Preventivo di periodo
6. Consuntivo di periodo
7. Retrospettiva

2.1.2.8 Piano di qualifica

Il piano di qualifica descrive gli obiettivi di qualità dei processi che il fornitore si impegna a soddisfare per consegnare un prodotto finale di qualità. Le metriche di valutazione vengono determinate dall'analisi dei requisiti e dalle indicazioni date dalla proponente, suddivise in base all'applicazione sui processi o sul prodotto. Le metriche stabilite vengono poi misurate attraverso opportuni test e verifiche, di cui vengono riportate le specifiche. Il documento include una sezione di rendicontazione per la valutazione dei processi e la valutazione del prodotto, in cui riportare l'attinenza alle metriche ottenuta rispetto agli obiettivi e di conseguenza valutare azioni correttive in caso si verifichino eventuali problemi (*cruscotto di qualità*$_G$). Il documento è diviso nelle seguenti sezioni:

1. Qualità dei processi
2. Qualità del prodotto
3. Specifiche di test e verifica
4. Cruscotto di qualità

2.1.2.9 Lettera di presentazione

La lettera di presentazione è il documento necessario alla candidatura per la milestone di revisione di avanzamento *RTB*$_G$ (*Requirements and Technology Baseline*). Essa contiene le informazioni sul repository di progetto, il puntatore al *Proof of Concept*$_G$, il consuntivo di spesa e preventivo a finire del progetto.

2.1.3 Strumenti

- *GitHub*$_G$ per la gestione della documentazione di progetto e mezzo comunicativo nella fase di fornitura
- *Jira*$_G$ per la suddivisione e il monitoraggio delle attività di progetto
- Discord per la comunicazione sincrona tra i membri del gruppo
- Gmail per la comunicazione asincrona con l'azienda proponente
- VSCode con estensione con IDE di riferimento.

2.2 Processo di sviluppo

Il processo di sviluppo prevede l'insieme di attività che definiscono lo svolgimento dell'Analisi dei Requisiti.

2.2.1 Attività di processo

L'elenco di attività previste è basato sullo standard ISO/IEC 12207:1995, e sono descritte di seguito:

- **Definizione del processo di sviluppo**: selezione del Ciclo di Vita del Software più idoneo tenendo conto degli obiettivi, della rilevanza e della complessità del progetto;
- **Raccolta e analisi dei requisiti**: attività volta a individuare e specificare le esigenze dell'utente finale rispetto alle funzionalità richieste al Software. Un'analisi esaustiva deve includere le funzioni del Sistema, i bisogni degli utilizzatori e i vincoli stabiliti dal committente;
- **Progettazione dell'architettura di Sistema**: identificazione delle componenti hardware e software necessarie a garantire il soddisfacimento di tutti i requisiti definiti, supportata da un adeguato tracciamento degli stessi;
- **Analisi dei requisiti Software**: studio di come il Software risponde ai requisiti lato utente, includendo anche gli aspetti di qualità, quali funzionalità (comprensive di requisiti prestazionali), interfacce tra componenti e requisiti di sicurezza;
- **Definizione dell'architettura Software**: progettazione delle principali componenti del sistema e delle loro interazioni, con particolare attenzione alla struttura complessiva piuttosto che ai dettagli implementativi;

- **Progettazione dettagliata del Software**: sviluppo del progetto delle singole componenti Software fino all'individuazione delle unità elementari;
- **Sviluppo e verifica del Software**: realizzazione delle unità che costituiscono le componenti progettate, accompagnata da test specifici per verificarne il corretto funzionamento;
- **Integrazione delle componenti Software**: assemblaggio delle diverse parti in componenti complete, supportato da test di integrazione per garantirne il comportamento corretto;
- **Test di qualificazione del Software**: esecuzione di test dedicati per verificare che il Software soddisfi i requisiti e gli obiettivi di qualità prefissati;
- **Integrazione del Sistema**: combinazione di tutte le componenti realizzate nel Sistema finale;
- **Test di qualificazione del Sistema**: verifica dell'intero Sistema attraverso test complessivi per accertarne il corretto funzionamento;
- **Installazione del Software**: consegna e messa in opera del prodotto presso il cliente finale nell'ambiente precedentemente concordato;
- **Supporto all'approvazione del Software**: attività di assistenza all'utente finale per verificare che tutti i requisiti richiesti siano stati effettivamente soddisfatti.

2.2.2 Analisi dei Requisiti

L'*Analisi dei Requisiti*$_G$ rappresenta una delle attività fondamentali all'interno della ***Requirements and Technology Baseline*** *(RTB)*$_G$ e ha l'obiettivo di identificare in modo completo l'insieme dei requisiti che il sistema sviluppato dovrà soddisfare.

I risultati di tale attività sono raccolti nel documento *Analisi dei Requisiti*, nel quale sono riportate in maniera dettagliata tutte le informazioni necessarie. Questo documento costituisce un riferimento essenziale per le successive fasi di progettazione dell'architettura e di codifica, supportando il lavoro dei progettisti e degli sviluppatori.

Un ulteriore elemento di riferimento è il *Piano di Qualifica* che, includendo l'elenco dei test e il loro stato di avanzamento, consente di verificare quali requisiti risultano soddisfatti e quali siano ancora da validare.

In particolare, il documento di Analisi dei Requisiti organizza i *casi d'uso*$_G$ individuati e i relativi requisiti associati. Al fine di agevolarne la consultazione, viene di seguito illustrata nel dettaglio la nomenclatura adottata.

2.2.3 Casi d'uso

I casi d'uso utilizzano la nomenclatura seguente:

UC-[Primario](-[Secondario]-[Terziario]-ecc.)

UC sta per Use Case, ovvero caso d'uso in inglese. Gli UC sono identificati univocamente tramite una numerazione crescente. *Primario* è il numero dello UC principale. È

possibile avere dei s*otto-UC*, in tal caso verrà aggiunto il numero crescente del sotto-UC come appendice allo UC genitore. Ogni UC è anche dotato di un nome descrittivo.

2.2.4 Requisiti

I requisiti utilizzano la nomenclatura seguente:

R-[ID]-[Tipo]-[Priorità]

Dove:

- **ID**: numero progressivo del requisito
- Tipo:
  - **F** (Requisiti Funzionali): descrivono le funzionalità del sistema
  - **Q** (Requisiti di Qualità): descrivono le caratteristiche qualitative del sistema
  - **V** (Requisiti di Vincolo): descrivono i vincoli tecnologici e normativi
- Priorità: **Ob** (Obbligatorio), **De** (Desiderabile), **Op** (Opzionale)

2.3 Processo operativo

Il processo operativo comprende quell'insieme di attività trasversali che sono necessarie a garantire il corretto coordinamento tra i membri del gruppo e il raggiungimento degli obiettivi del progetto. Tale processo assicura comunicazioni efficaci, nonchè una distribuzione ottimale dei compiti, al fine di mantenere alta la qualità del software e rispettare le tempistiche stabilite. Il processo operativo si integra naturalmente con tutti gli altri processi del progetto.

2.3.1 Pianificazione operativa

La pianificazione operativa rappresenta l'organizzazione delle attività quotidiane e settimanali del gruppo. Il Responsabile, all'inizio di ogni sprint, coordina la distribuzione dei compiti tra i membri.

Procedure di pianificazione operativa:

1. Svolgimento di una riunione di pianificazione all'inizio di ogni sprint;
2. Discussione degli obiettivi dello sprint e le attività necessarie per raggiungerli;
3. Assegnamento delle task ai vari membri;
4. Scelta delle scadenze intermedie.

2.3.2 Gestione dei rischi operativi

La gestione dei rischi operativi mira a identificare in maniera proattiva le potenziali problematiche che potrebbero minare il normale svolgimento delle attività e a pianificare le dovute azioni di mitigazione. I principali rischi operativi identificati sono:

- Sforamento dei costi preventivati;
- Calo di produttività del team;
- Mancata comunicazione e collaborazione tra i membri del team;
- Mancata comunicazione con l'azienda proponente;
- Problemi tecnici con gli strumenti di sviluppo;
- Mancato rispetto delle norme e documenti di progetto interni.

Le strategie adottate per ogni tipologia di rischio sono rispettivamente:

- Monitoraggio costante dell'allocazione delle ore rispetto alla pianificazione iniziale, svolgimento di stand-up meetings periodici e previsione margini temporali per imprevisti;
- Pianificazione anticipata delle attività più critiche prima del periodo di calo, e le restanti tenendo conto del periodo di ridotta attività;
- Adozione di canali di comunicazione chiari e regolari e una routine di aggiornamenti pianificati per garantire che tutti i membri del team siano allineati sugli obiettivi e le responsabilità;
- Scelta di un calendario di incontri regolari con l'azienda proponente per garantire un flusso costante di comunicazione e feedback;
- Impostazione di sessioni di formazione iniziali con il supporto occasionale dell'azienda proponente per familiarizzare con gli strumenti e le tecnologie;
- Ruolo attivo di amministratori e tester per garantire il rispetto delle norme e dei documenti di progetto interni.

2.4 Processo di manutenzione

Il processo di manutenzione definisce le modalità con cui il gruppo gestisce le modifiche, le correzioni e gli aggiornamenti del software e della corrispondente documentazione durante tutto il ciclo di vita del progetto. Questo processo garantisce che il prodotto rimanga conforme ai requisiti. Eventuali difetti devono essere corretti tempestivamente mentre, ove sorgesse la necessità di apportare migliorie, queste ultime vengano fornite in modo efficace ed efficiente.

2.4.1 Manutenzione correttiva

Si tratta di interventi finalizzati alla correzione di difetti o malfunzionamenti trovati nel software o nella documentazione. Sono inclusi:

- Correzioni di bug;
- Risoluzione di errori di varia natura nella documentazione;
- Correzione di qualsiasi comportamento non conforme ai requisiti.

2.4.2 Manutenzione adattiva

Sono le modifiche necessarie per adattare il prodotto a cambiamenti in itinere nell'ambiente operativo o nei requisiti. Sono inclusi:

- Adattamento a nuove tecnologie;
- Modifiche conseguenti a variazioni nei requisiti.

2.4.3 Manutenzione preventiva

Si risferisce alle attività proattive per ridurre la probabilità di problemi futuri. Sono inclusi:

- Aggiornamenti di sicurezza;
- Miglioramento della gestione degli errori.

2.4.4 Identificazione della necessità di manutenzione

La necessità di manutenzione può emergere da diverse fonti, come segnalazioni interne quali le attività di verifica e validazione, lo sviluppo del codice e le retrospettive; o segnalazioni esterne quali feedback della proponente.

2.4.5 Verifica e validazione

La verifica va effettuata rigorosamente prima dell'integrazione. Il verificatore assegnato deve esaminare il codice modificato verificando la conformità agli standard e l'assenza di introduzione di problematiche note. Deve inoltre eseguire i test, verificare che non siano presenti regressioni e controllare che la documentazione sia stata aggiornata coerentemente.

La validazione riguarda modifiche significative che possono impattare i requisiti o l'architettura. Tali modifiche vengono presentate illustrando il problema e la soluzione implementata e si accoglie il feedback della proponente.

2.4.6 Metriche

Per valutare l'efficacia della manutenzione, il gruppo monitora le seguenti metriche:

- Tempo medio di risoluzione: tempo medio tra l'identificazione di un problema e la sua risoluzione;
- Percentuale di regressioni: numero di nuovi difetti introdotti da attività di manutenzione rispetto al totale delle modifiche;
- Distribuzione per tipologia: suddivisione degli interventi di manutenzione per categoria.

3 Processi di Supporto

I processi di supporto sono volti a garantire l'efficacia e l'efficienza dei processi primari.

3.1 Processo di documentazione

Il processo di documentazione è parte integrante del Progetto in quanto permette il tracciamento delle decisioni prese, delle attività svolte e dei risultati ottenuti. Tutto ciò al fine di favorire il lavoro asincrono tra membri del gruppo e promuovere il principio *Agile*$_G$ di continuo miglioramento e adattamento tramite *feedback*$_G$.

3.1.1 Attività: Pianificazione della documentazione

La pianificazione della documentazione avviene contestualmente alla pianificazione delle attività di progetto.
Durante la pianificazione di ogni *sprint*$_G$, il *responsabile di progetto*$_G$ assegna le attività di documentazione ai membri del gruppo, tenendo conto delle competenze e della disponibilità di ciascuno. Le scadenze per la consegna dei documenti sono stabilite in modo da garantire che la documentazione sia sempre aggiornata e disponibile per la consultazione da parte del gruppo e di eventuali attori esterni (Azienda *proponente*$_G$, *committente*$_G$).
Per una più efficiente scrittura dei documenti, soprattutto di tutti quei documenti periodici (Verbali Interni, Verbali Esterni, Diario di Bordo) sono presenti modelli standard approvati in /assets. L'aggiornamento di tali standard deve essere argomento di Verbali Interni e risultato di una discussione e successiva decisione presa in tale sede.

3.1.1.1 Procedure di Pianificazione

I seguenti passaggi guidano il Team nella pianificazione delle attività di documentazione:

1. Durante la pianificazione di ogni *sprint*$_G$, il *responsabile*$_G$ identifica le necessità di documentazione in base agli obiettivi dello sprint e alle attività previste.

2. Il responsabile assegna le attività di documentazione ai membri del gruppo, tenendo conto delle competenze e della disponibilità di ciascuno, nonchè della necessità di ruotare i ruoli, per dare la possibilità a tutti i membri di acquisire esperienza in diverse aree.
3. Vengono create le *issue*$_G$ in Jira per ogni attività di documentazione, specificando i dettagli del compito, le scadenze e i verificatori per ogni attività. Specifiche in 3.2.1.1.

3.1.1.2 Strumenti di Pianificazione

- *Jira*$_G$ per la gestione delle attività di progetto. In particolare con la *board*$_G$ *Scrum*$_G$ che viene aggiornata in automatico con i commit effettuati sui Work Item e può essere personalizzata con la creazione di *Sprint*$_G$.
- *DashBoard/Cruscotto*$_G$ di Jira per il monitoraggio delle attività assegnate per ogni membro del gruppo.

3.1.2 Attività: Produzione della documentazione

La produzione della documentazione è un'attività fondamentale per garantire la tracciabilità e la comunicazione efficace all'interno del gruppo e con gli stakeholder esterni. Ogni documento deve essere redatto seguendo le linee guida stabilite, utilizzando i modelli approvati e rispettando le scadenze stabilite durante la pianificazione.

3.1.2.1 Linee guida per la produzione della documentazione

Per garantire la coerenza e la qualità della documentazione, il gruppo segue le seguenti linee guida:

- **Verbali:** la stesura dei verbali utilizza un template standard approvato per garantire continuità e chiarezza nella comunicazione. La struttura prevede:
  - Intestazione con logo del gruppo, data, durata della riunione e luogo (fisico o virtuale);
  - Indice dei contenuti;
  - Tabella di versionamento;
  - elenco dei partecipanti;
  - Ordine del giorno;
  - Resoconto dettagliato dei punti discussi;
  - Tabella di definizione dei ruoli (se necessario);
  - Decisioni prese e tabella delle attività associate.
- **Diario di bordo:** la struttura dei diari di bordo segue anch'essa un template standard che include il logo del gruppo, la data, un numero progressivo, e il numero del gruppo. Le sezioni principali sono:

  - Difficoltà affrontate;
  - Dubbi e incertezze;
  - Pagina conclusiva.

- **Altri documenti:** Per tutti gli altri documenti non vengono utilizzati modelli particolari in quanto prevediamo una sola istanza di ciascuno di essi. Tuttavia seguiamo delle linee guida comuni per garantire consistenza dell'impianto tipografico. In particolare:
  - Frontespizio con logo e titolo del documento;
  - Tabella di versionamento;
  - Indice dei contenuti.

3.1.2.1.1 Denominazione e datazione documentazione

Per una corretta archiviazione e reperibilità delle modifiche apportate ai documenti, è necessario seguire le seguenti convenzioni per la datazione e denominazione.

- Tutti i file dei documenti devono seguire la regola del **Pascal Case**, ovvero devono essere scritti senza spazi e con la prima lettera di ogni parola in maiuscola.
- All'interno dei documenti, la data deve essere riportata nel formato **YYYY-MM-DD**, anno-mese-giorno.

La denominazione dei file sul sito segue anch'essa la datazione usata per il nome dei file su GitHub ovvero **YYYY-MM-DD**, con l'aggiunta del numero di versione alla fine del nome. Per le specifiche di versionamento, si rimanda alla sezione 3.2.4.1.

3.1.2.2 Strumenti di Produzione

- *VSCode*$_G$ come IDE principale per la stesura dei documenti in *LaTeX*$_G$.
- *Estensione Jira per VSCode*$_G$ per la gestione dei Work Item assegnati e la creazione automatica delle branch di lavoro.
- *GitHub*$_G$ per la gestione delle versioni della documentazione di progetto.

3.1.3 Attività: Revisione e Approvazione

Ogni documento redatto viene sottoposto a un processo di revisione interna che ne accerta la correttezza contenutistica, formale, e stilistica. La revisione viene effettuata da un membro del gruppo diverso dall'autore del documento seguendo la procedura definita a seguire

3.1.3.1 Procedure di Revisione e Approvazione

1. Una volta ricevuta la notifica della PR da revisionare, il revisore dovrà controllare l'aderenza ai modelli approvati, la correttezza formale e sostanziale del documento. Per velocizzare, oltre alla lettura attenta, si consiglia l'uso di LLM, in particolare per l'analisi grammaticale e stilistica.

2. Completata la revisione, il revisore può:

   - **Approvare la PR**, notificando all'autore l'approvazione. È necessario che nel testo del commit del merge siano chiuse tramite Smart Commit entrambe le issue correlate.
   - **Richiedere modifiche**, fornendo un feedback dettagliato all'autore, chiudendo la PR che sarà riaperta dall'autore una volta implementate le modifiche.

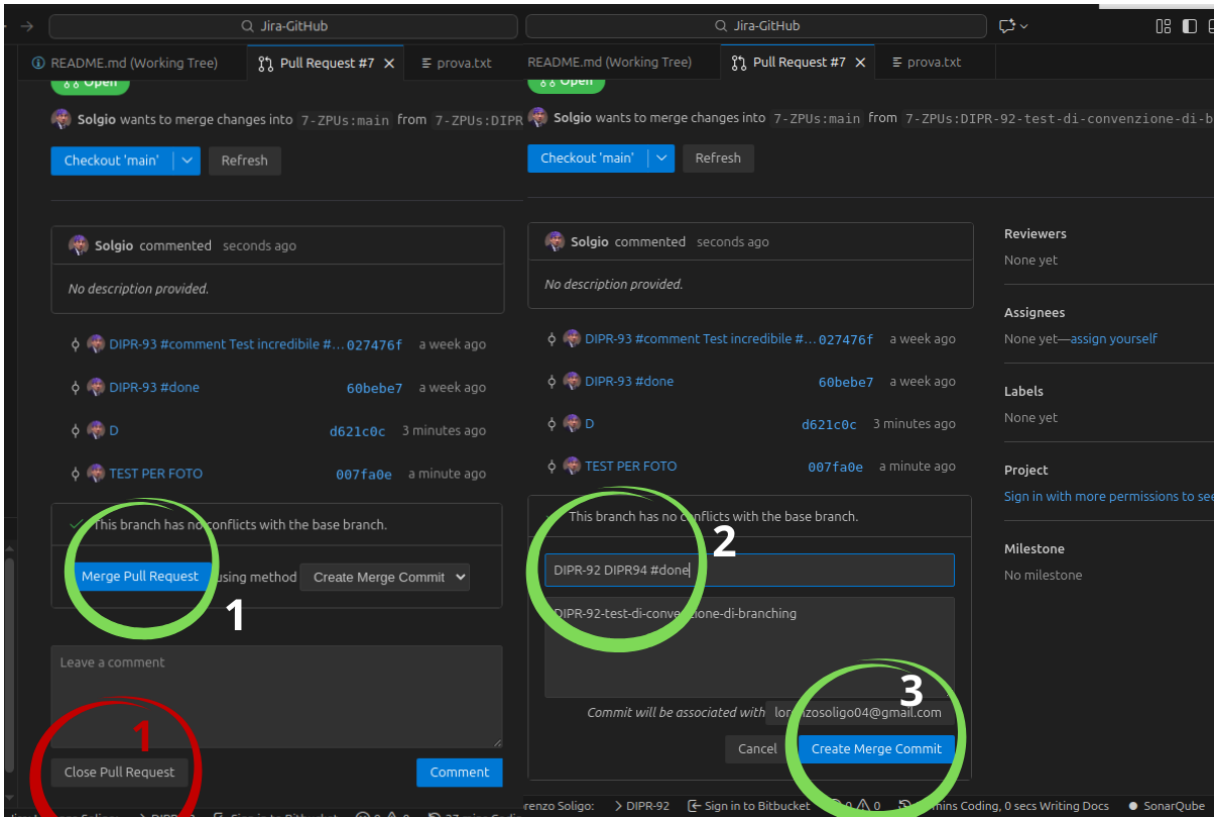

Figura 1: Commit della PR con Smart Commit verso le due issue

Per quanto riguarda l'approvazione finale del documento, questa spetta al *responsabile*$_G$, il quale effettua il merge da "*in_lavorazione*" nel ramo principale *main*, base per la versione ufficiale di rilascio corrispondente alla *milestone*$_G$. Questo passaggio dovrebbe risultare puramente formale. Non di meno è il garante delle qualità del documento quindi deve impiegare il proprio tempo, il minimo possibile, per rileggere e confermare i contenuti.

3.1.3.2 Strumenti di Revisione e Approvazione

Per la gestione della documentazione di progetto il gruppo utilizza i seguenti strumenti:

- *GitHub*$_G$, in particolare integrato in VSCode per la gestione delle versioni e delle pull request, comodamente nell'ambiente di sviluppo di VSCode 4.4.0.3. Per una stesura efficiente dei documenti il Team si è dotato di modelli predefiniti (Decisione del 2025-11-07)·
- *Jira*$_G$: strumento di gestione delle attività di progetto, utilizzato per tracciare le attività di documentazione e assegnarle ai membri del gruppo.
- *LLM*$_G$ per il supporto alla revisione formale e stilistica dei documenti.

3.2 Processo di Gestione della Configurazione

Il processo di gestione della configurazione ha lo scopo di identificare, definire e controllare gli elementi della configurazione software (*SCIs*$_G$) durante tutto il ciclo di vita del progetto, garantendo la tracciabilità delle modifiche e l'integrità dei rilasci. Gli SCIs in poche parole sono tutti gli artefatti prodotti e gestiti durante il progetto.

3.2.1 Attività: Identificazione della Configurazione

Questa attività prevede la definizione degli elementi della configurazione (SCIs) e la loro identificazione univoca.

3.2.1.1 Procedure: Identificazione degli SCIs

Gli SCIs sono sempre associati ad un Work Item di Jira, che ne garantisce la tracciabilità e la gestione delle modifiche. In particolare il processo di creazione deve seguire i seguenti passaggi:

1. Creazione di una issue in Jira per ogni nuovo artefatto da produrre (documento, componente software, etc).
2. Identificazione dell'ambito di appartenenza (Epic), della funzionalità (Feature) o di una specifica attività (Task) già presenti nel sistema o da aggiungere se non compatibile.
3. Assegnazione della issue al membro del gruppo responsabile della sua supervisione.
4. Aggiunta di label specifici per facilitare la ricerca e la categorizzazione degli SCIs, per esempio *DOCS*, *Formazione*, *Code* etc.
5. Aggiunta di Linked Issues per collegare SCIs correlati o dipendenti tra loro, con relazioni di *child/parent of* o *blocked by* per esempio.
6. Aggiunta di eventuali allegati.

7. Definizione delle scadenze e del *Time Estimate*$_G$ per la gestione del carico di lavoro.

È quindi necessaria una specificazione sulla struttura di un Work Item. Ogni Work Item deve contenere deve consistere in una fase produttiva e una fase di revisione, per garantire la qualità del prodotto finale.
A tal fine si adottano le seguenti convenzioni:

- Ogni Work Item deve presentare una sotto issue che rappresenta la fase di produzione.
- La sotto issue di produzione deve essere collegata alla issue principale tramite la relazione *child of*.
- La sotto issue di produzione deve essere assegnata al membro del gruppo responsabile della stesura o sviluppo dell'artefatto, mentre la issue principale deve essere assegnata al membro responsabile della supervisione.

Questo approccio divide chiaramente le responsabilità e le attività, mantenendo la correlazione tra produzione e supervisione e inoltre facilita il monitoraggio dello stato di avanzamento, la gestione delle revisioni e conteggio di *Ore produttive*$_G$ consumate. Di più nello specifico in **??**.

3.2.1.1.1 Issue e SubIssue

In generale, come già detto, ogni Work Item deve essere composto da una issue principale e una sotto issue di produzione. Nel caso però di Riunioni e Diari di Bordo, intesi come eventi ma anche come tipi di work item Jira, la struttura cambia:

- **Riunioni:** La Riunione rappresenta l'evento, simile al concetto di Epic o Feature, mentre le issue rappresentano i verbali interni ed esterni associati alla riunione stessa.
- **Diari di Bordo:** Similarmente, il Diario di Bordo rappresenta l'evento periodico, mentre la issue associata riguarda la preparazione delle slide.

Per entrambi, la creazione della issue di produzione avviene automaticamente una volta che l'evento viene assegnato al membro del team che si occuperà della verifica del materiale prodotto.

3.2.1.2 Strumenti di Identificazione

- **Jira:** Strumento di gestione delle attività di progetto.

3.2.2 Attività: Controllo della configurazione

Il controllo della configurazione è il processo con il quale vengono gestite le richieste di modifica. Una volta effettuata una modifica l'autore crea una *(PR) Pull Request*$_G$ verso la feature branch principale, assegnando come revisore il membro del gruppo designato, diverso da se.
A questo punto:

- Se il revisore **approva la PR**, questo branch viene automaticamente eliminato, il work item viene marcato come completato in Jira e l'assegnatario può proseguire con gli altri compiti a lui assegnati.
- Se il revisore richiede modifiche **la PR viene rifiutata** e l'assegnatario deve procedere con le modifiche richieste. Una volta completate, l'assegnatario notifica il revisore che procederà con una nuova revisione. Questo ciclo si ripete fino a quando la PR non viene approvata.

L'integrazione con Jira permette di controllare lo stato di avanzamento dei Work Item, la rendicontazione delle ore lavorate e la gestione delle scadenze. Risulta quindi **obbligatorio** l'utilizzo di Smart Commit per tutti i commit, compresi quelli di Pull Request. Più in 3.2.2.1.

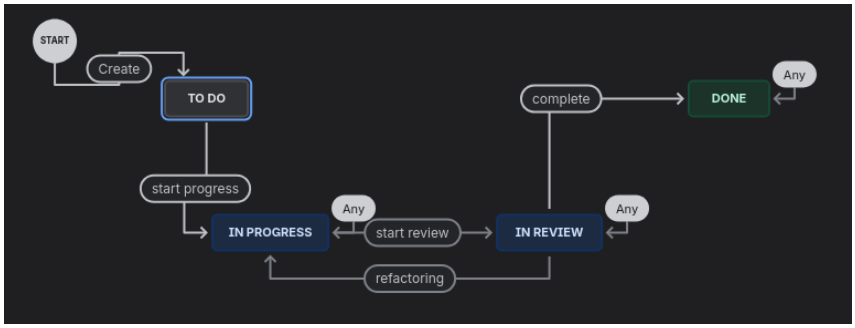

Figura 2: Flusso del processo di controllo

I seguenti passaggi guidano il Team nella produzione di modifiche alla baseline:

1. Consultando l'estensione "Atlassian: Jira, Rovo Dev, Bitbucket" il membro del gruppo potrà avere accesso al Work Item assegnatogli e cliccando su "Start Work" potrà **creare il branch** di lavoro secondo le convenzioni stabilite (3.2.3.1). Nell'immagine sottostante sono indicati visivamente i passi.

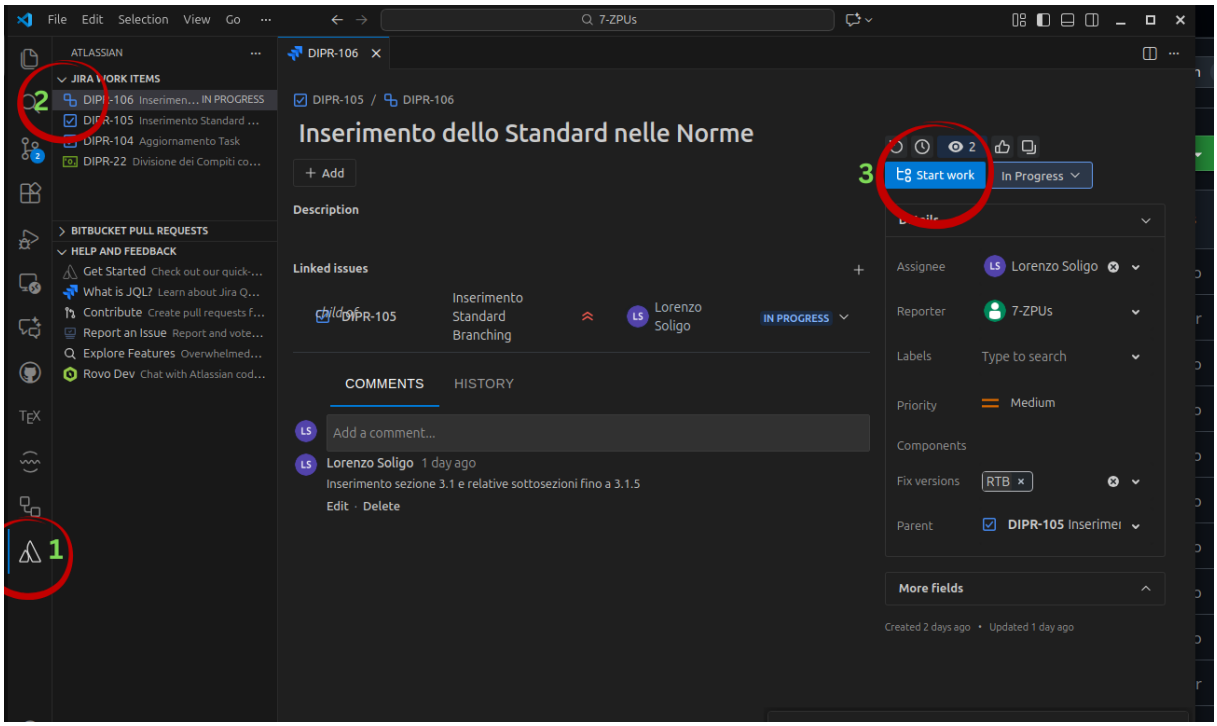

Figura 3: Creazione del branch di lavoro tramite estensione Jira in VSCode

2. Quando è necessario **effettuare un commit** è obbligatorio utilizzare lo Smart Commit. Più in 4.4.0.2.

3. Una volta completata la task, è necessario **creare la PR** verso la branch *"in_lavorazione"* di riferimento mettendo come revisore il membro del gruppo designato. Per esempio, una volta completata la stesura di un verbale viene creata una PR verso *verbali_in_lavorazione* e se questa viene approvata, il documento sarà pronto per la revisione finale del responsabile che si occuperà del merge al *main* in sede di milestone.

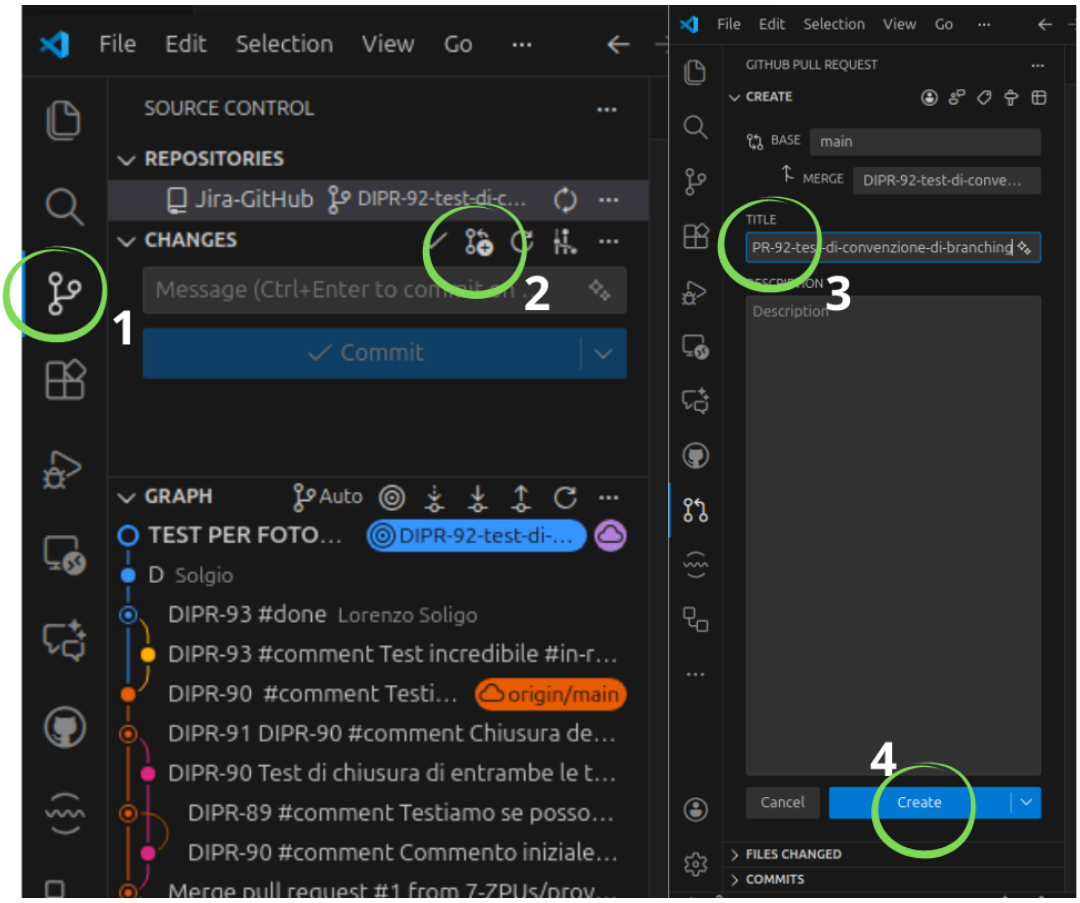

Figura 4: Creazione della PR verso l'issue branch *in_lavorazione*

3.2.2.1 Procedure: Smart Commit

La necessità di tracciamento delle attività richiede l'adozione capillare degli Smart Commit per collegare i commit GitHub alle issue Jira. Questo permette di aggiornare automaticamente lo stato delle task e rendicontare il tempo.

La sintassi obbligatoria è:

```
DIPR-<numero issue> #time <nd nh nm> #<stato> #comment <Descrizione>
```

Regole di applicazione:

1. **Time Tracking:** L'inserimento del tempo (`#time`) deve seguire il formato *nd nh nm* (giorni, ore, minuti).
2. **Transizioni di Stato:** Utilizzare i tag `#start-progress`, `#start-review`, `#complete` per avanzare il workflow su Jira.
3. **Pull Request:** Nelle PR è vietato inserire il tag `#time` per evitare la doppia contabilizzazione delle ore lavorative.
4. **Issue Multiple:** È possibile agire su più issue in un singolo commit separandole con spazi (utile per chiudere issue di sviluppo e verifica contemporaneamente)[cite: 97, 98].

3.2.2.2 Strumenti di Controllo delle Configurazioni

- **Jira Automation:** Interpreta gli Smart Commit per aggiornare i Work Item.
- **Git/GitHub:** Motore di versionamento sottostante.

3.2.3 Attività: Versionamento e Identificazione

Questa attività definisce le regole per l'identificazione univoca degli artefatti e la gestione delle ramificazioni nel repository.

3.2.3.1 Procedure: Standard per le Branch

Per garantire una gestione ordinata e coerente del codice e della documentazione, il Team adotta il seguente standard per la denominazione dei branch in GitHub:

- La creazione del branch deve avvenire preferibilmente in modo automatico tramite l'integrazione Jira-VSCode 4.4.0.3.
- Il formato obbligatorio è:

```
DIPR-<numero issue>-<descrizione-breve>
```

- È necessario scegliere una descrizione breve che identifichi chiaramente il contenuto della modifica.

3.2.3.2 Strumenti di Versionamento

- **GitHub:** Repository remoto per la memorizzazione delle versioni.
- **VSCode (Estensione Jira):** Per l'automazione della nomenclatura delle branch.

3.2.4 Registrazione delle Configurazioni

Questa attività prevede la documentazione e la registrazione delle modifiche apportate agli SCIs.

3.2.4.1 Procedure: Registrazione delle Modifiche

Per ogni modifica apportata a uno SCI, è necessario documentare le seguenti informazioni, all'interno della sezione denominata "Tabella di Versionamento", situata all'inizio di ogni documento.
Al suo interno vi sono:

- Numero di versione
- Data della modifica (Di più in 3.1.2.1.1)

- Autore della modifica
- Verificatore della modifica
- Descrizione della modifica, con riferimento preciso, al paragrafo o sezione interessata.

Per la numerazione delle versioni si adotta lo standard di versionamento **MAJOR.MINOR.PATCH**, nel quale:

- **MAJOR** viene incrementato per modifiche sostanziali che introducono cambiamenti significativi o incompatibili con le versioni precedenti. Viene modificato in vista delle milestone.
- **MINOR** viene incrementato per l'aggiunta di nuove funzionalità o miglioramenti che non compromettono la compatibilità con le versioni precedenti.
- **PATCH** viene incrementato per correzioni di bug, miglioramenti minori o modifiche che non influenzano le funzionalità principali del documento.

3.3 Processo di verifica

La verifica serve a controllare che il software prodotto corrisponda ai requisiti richiesti dalla proponente. Per evitare problemi, a ogni task viene assegnato uno o più verificatori. A seconda del tipo di task, i verificatori useranno tecniche e pratiche adeguate per verificare che il prodotto rispetti gli standard di qualità. Dentro il Piano di Qualifica sono elencate le attività che i verificatori devono fare al completamento della task a loro assegnatali. Troviamo anche per ogni attività gli obbiettivi, gli esiti desiderabili e quelli ottenuti.

3.3.1 Verifica documentale

I passaggi che il verificatore deve svolgere per approvare un documento sono specificati nella sezione 3.1.2. Perché un documento possa essere approvato il verificatore deve controllare che sia corretto:

- Il contenuto.
- La grammatica.
- La presentazione, rispetto alle linee guida.
- La comprensibilità del testo.

Individuiamo due tipologie di verifica inerenti al codice, l'analisi statica e quella dinamica

3.3.2 Analisi Statica

L'analisi statica prevede di analizzare il codice in modo statico, ovvero senza che venga eseguito. Ci concentriamo sulla documentazione e sul codice, e ci accertiamo che non presentino errori che non li rendano conformi alle norme, o che non producano risultati attesi. Per fare ciò possiamo utilizzare dei metodi di lettura oppure dei metodi formali. Iniziamo concentrandoci sui metodi di lettura, possono essere svolti da esseri umani o tramite processi automatizzati, e si suddividono in:

- Walkthrough: a priori non conosciamo l'errore, ne sappiamo la sua locazione, quindi andiamo ad analizzare il testo in maniera esplorativa. I verificatori si occupano di quest'ultimo passo, la correzione sarà poi a carico degli autori del testo in questione. Questo processo potrebbe risultare particolarmente laborioso in proporzione alla quantità di dati da ispezionare.
- Inspection: sappiamo dove si potrebbe situare l'errore e quindi a nostro vantaggio possiamo essere selettivi rispetto alle porzioni di testo da esaminare. Come per il Walkthrough sono gli autori a correggere l'errore, mentre i verificatori lo cercano. Questo tipo di metodo ha come presupposto la conoscenza del problema a priori, questa conoscenza è ottenibile tramite esperienza.

3.3.3 Analisi Dinamica

L'analisi dinamica consiste nell'eseguire l'oggetto da verificare. Durante l'esecuzione si possono eseguire dei test, che devono essere **ripetibili** e **automatizzabili**. I test devono essere ripetibili per evitare regressioni del software e automatizzati per essere più efficienti. Per farlo si usano:

- **Driver**, che permette di invocare il componente da testare se ancora nessuno lo chiama;
- **Stub**, che evita le chiamate a moduli non testati;
- **Logger**, che traccia i test.

Vediamo le tipologie di test più comuni:

- **Unità**: verificano il singolo componente in isolamento. Si suddividono in funzionali e strutturali, i primi si basano sulle specifiche funzionali e non sull'implementazione, cioè verificano il comportamento dell'unità in base alle specifiche, i secondi invece sulla struttura interna del codice, come ad esempio la copertura dei rami.
- **Regressione**: verificano che modifiche o correzioni non introducano nuovi errori in parti del codice già verificate. Ripetiamo i test già eseguiti.
- **Integrazione**: verifichiamo che più unità già testate singolarmente possano essere integrate, cioè interagiscano correttamente.
- **Sistema**: verificano la totalità del sistema rispetto ai requisiti. Andiamo a collaudare il sistema con il committente, questo fa parte del processo di validazione, trattato nella sezione successiva

3.4 Processo di validazione

La validazione serve a verificare se i requisiti e le aspettative della proponente sono stati soddisfatti tramite test di accettazione. Questi test simulano uno scenario realistico, permettendo di mostrare il prodotto alla proponente e raccogliere il suo feedback.

3.5 Processo di revisione congiunta

Il processo di revisione congiunta regola le attività di verifica periodica svolte congiuntamente con la proponente per verificare l'allineamento rispetto ai requisiti e agli obiettivi concordati. Questo processo garantisce che lo sviluppo del prodotto proceda in conformità con le aspettative della proponente e permette di identificare tempestivamente eventuali problemi.

I ruoli coinvolti in questo processo sono:

Ruolo	Specifica
Responsabile	Coordina gli incontri preparando i materiali e gestendo la comunicazione con la proponente
Tutti i membri del team	Partecipano attivamente alle revisioni ponendo quesiti quando necessario

3.5.1 Attività: Pianificazione delle revisioni

3.5.1.1 Procedure di Revisione

Come concordato con la proponente (2.1), con cadenza bisettimanale il gruppo di lavoro si incontra con l'azienda tramite Google Meet per discutere lo stato di avanzamento del progetto.
L'amministratore si occupa della creazione della task Jira relativa all'incontro, assegnando anche la scrittura e verifica del verbale.

3.5.2 Attività: Preparazione dei materiali

3.5.2.1 Procedure di preparazione

Al fine di garantire un'organizzazione efficace degli incontri, vengono adottate le seguenti modalità operative:

1. È già stato fornito alla proponente un link alla dashboard per il monitoraggio continuo dell'avanzamento dei lavori.
2. Prima di ogni incontro si svolge un meeting su Discord per definire l'ordine di esposizione degli argomenti.
3. Prima di ogni incontro viene concordato il materiale da mostrare durante la revisione. Vengono, inoltre, raccolte e organizzate le domande, provenienti da tutti i membri del gruppo, da sottoporre alla proponente.

3.5.3 Attività: Conduzione della revisione

3.5.3.1 Procedure di conduzione

Durante l'incontro di revisione congiunta:

1. Il responsabile presenta lo stato di avanzamento rispetto alla pianificazione
2. Vengono mostrate, se presenti, le demo delle nuove funzionalità implementate
3. Si discutono eventuali problematiche emerse e si raccolgono feedback dalla proponente chiarendo eventuali dubbi

3.5.4 Attività: Post incontro

3.5.4.1 Procedure post incontro

Al termine di ogni revisione congiunta:

1. Viene completato il verbale esterno riportante le tematiche trattate e i dubbi discussi
2. Il verbale viene condiviso con la proponente tramite Google Drive per approvazione e firma
3. Eventuali variazioni rispetto alla visione iniziale del gruppo vengono formalizzate tramite la creazione di apposite Issue Jira e assegnate al membro del team competente.
4. Il verbale approvato viene archiviato nel repository secondo le procedure di documentazione (3.1.2)

3.5.5 Strumenti di revisione congiunta

- **Google Drive:** Per la condivisione dei verbali esterni e dei materiali di revisione con la proponente
- **Gmail:** Per la comunicazione formale con la proponente
- **Discord:** Per il coordinamento interno del team in preparazione alle revisioni
- **Jira:** Per il tracciamento delle Issue emerse dalle revisioni
- **Google Meet:** Per lo svolgimento delle revisioni in modalità sincrona con la proponente

3.6 Processo di risoluzione dei problemi

Il processo di risoluzione dei problemi definisce le modalità sistematiche per identificare, analizzare e risolvere problematiche tecniche oppure organizzative che emergono durante lo sviluppo.
Questo processo garantisce che i problemi vengano gestiti in modo strutturato, documentato e tempestivo, minimizzando l'impatto sulle attività di progetto.

I ruoli coinvolti in questo processo sono:

Ruolo	Specifica
Responsabile	Coordina l'analisi dei problemi critici e decide l'allocazione delle risorse per la risoluzione
Amministratore	Gestisce i problemi legati all'infrastruttura e agli strumenti, come descritto in 4.2
Analista/Progettista	Risolve problemi legati a requisiti ambigui o scelte architetturali
Programmatore	Risolve problemi tecnici nel codice e implementa le soluzioni concordate
Verificatore	Verifica che le soluzioni implementate risolvano effettivamente il problema

3.6.1 Attività: Identificazione del problema

3.6.1.1 Procedure di identificazione

Qualsiasi membro del team può identificare e segnalare un problema utilizzando uno dei seguenti canali di comunicazione:

1. Gruppo WhatsApp, per segnalazioni rapide e comunicazioni immediate.
2. Server Discord, per discussioni strutturate e confronti approfonditi.
3. Chiamate straordinarie tra sottogruppi di membri, in caso di problematiche che richiedano un confronto diretto.
4. Riunione settimanale del team, per l'analisi condivisa di problematiche di carattere organizzativo, tecnico o progettuale
5. Stand-up meeting periodici

La segnalazione di un problema deve includere, ove possibile, le seguenti informazioni:

- Descrizione chiara e dettagliata del problema riscontrato.
- Indicazione del livello di urgenza e della priorità suggerita.
- Eventuali tentativi di risoluzione già effettuati o possibili soluzioni proposte.

3.6.2 Attività: Analisi e classificazione

3.6.2.1 Procedure di analisi e classificazione

Una volta identificato, il problema viene analizzato dal responsabile insieme ai membri competenti per:

1. **Classificare la tipologia:**
    - Problema tecnico
    - Problema di processo
    - Problema organizzativo
2. **Valutare la priorità:**
    - *Critica:* Blocca completamente le attività, richiede risoluzione immediata
    - *Alta:* Impatta significativamente il progresso
    - *Media:* Causa rallentamenti, da risolvere entro lo sprint corrente
    - *Bassa:* Impatto minimo, da risolvere quando possibile
3. **Identificare le cause radice:** Attraverso tecniche di analisi
4. **Stimare risorse da allocare:** Per la risoluzione del problema

3.6.3 Attività: Sviluppo della soluzione

3.6.3.1 Procedure di sviluppo della soluzione

In base alla classificazione e all'analisi effettuata:

1. Il responsabile assegna il problema al membro o ai membri più competenti
2. Viene creata una issue Jira collegata al problema
3. Il team propone una o più soluzioni alternative, valutando:
    - Efficacia nel risolvere il problema
    - Impatto su altre componenti o processi
    - Tempo e risorse necessarie
    - Rischi associati all'implementazione

3.6.4 Attività: Implementazione e verifica

3.6.4.1 Procedure di implementazione e verifica

La soluzione viene implementata seguendo le normali procedure di sviluppo:

1. Creazione del branch di lavoro secondo gli standard (3.2.3)

2. Implementazione della soluzione con commit e smart commit (3)
3. Verifica della soluzione da parte di un verificatore diverso dall'implementatore, mediante l'esecuzione di test volti ad accertare che la soluzione risolva effettivamente il problema.
4. Merge della soluzione secondo le procedure di revisione (3.1.3)

3.6.5 Attività: Procedure generali di risoluzione

3.6.5.1 Procedure generali di risoluzione

Le seguenti procedure guidano il Team nella risoluzione sistematica dei problemi:

1. **Segnalazione immediata:** Ogni problema deve essere segnalato appena identificato, non posticipato
2. **Comunicazione trasparente:** I problemi critici vengono comunicati immediatamente a tutto il team e, se necessario, alla proponente
3. **Documentazione completa:** Ogni problema risolto deve essere documentato per evitare ricorrenze

3.6.6 Integrazione con altri processi

Il processo di risoluzione dei problemi si integra con:

- **Gestione dei rischi operativi (2.3.2):** I problemi ricorrenti vengono aggiunti al registro dei rischi
- **Analisi degli incidenti (4.5.1):** Per problemi infrastrutturali seguono le procedure specifiche
- **Processo di miglioramento (4.5):** I problemi sistemici alimentano i cicli di miglioramento continuo
- **Metriche di qualità:** Le metriche aiutano a identificare aree problematiche in modo proattivo

3.6.7 Strumenti di risoluzione dei problemi

- **Jira:** Per il tracciamento formale dei problemi, della loro classificazione e risoluzione
- **Discord:** Per la comunicazione immediata di problemi urgenti e per sessioni di problem solving sincrone
- **Whatsapp:** Per notifiche urgenti di problemi critici che richiedono attenzione immediata
- **Dashboard Jira:** Per il monitoraggio dello stato dei problemi aperti e delle tendenze nel tempo
- **GitHub:** Per la gestione dei branch di lavoro e il merge delle soluzioni implementate

3.7 Gestione della qualità

Il processo di gestione della qualità sovrintende all'applicazione sistematica delle metriche definite e all'integrazione tra i processi di garanzia della qualità, verifica, validazione e miglioramento per garantire la qualità complessiva del prodotto finale e dei processi. Questo processo coordina tutte le attività volte a mantenere e migliorare gli standard qualitativi durante l'intero ciclo di vita del progetto.

I ruoli coinvolti in questo processo sono:

Ruolo	Specifica
Responsabile	Coordina le attività di gestione della qualità e garantisce il rispetto degli obiettivi definiti nel Piano di Qualifica
Verificatori	Eseguono le verifiche secondo le procedure definite e monitorano le metriche di qualità di processo e prodotto
Amministratore	Configura e mantiene gli strumenti di automazione per il monitoraggio della qualità (GitHub Actions, Dashboard e Issue Jira)
Tutti i membri	Contribuiscono al mantenimento della qualità seguendo le norme e segnalando deviazioni dagli standard

3.7.1 Attività: Pianificazione della qualità

3.7.1.1 Procedure di pianificazione

All'inizio di ogni sprint, il responsabile coordina la pianificazione degli obiettivi di qualità:

1. **Definizione degli obiettivi di sprint:** In base alle attività pianificate, vengono identificati gli obiettivi specifici di qualità da raggiungere
2. **Selezione delle metriche applicabili:** Tra quelle definite nella sezione 5, vengono scelte le metriche più rilevanti per lo sprint corrente
3. **Definizione delle soglie:** Per ogni metrica selezionata, vengono stabilite le soglie minime accettabili e quelle target, in conformità con il Piano di Qualifica
4. **Assegnazione delle responsabilità:** I verificatori vengono assegnati alle diverse attività di controllo

Gli obiettivi di qualità vengono documentati e comunicati a tutto il team durante la riunione di pianificazione dello sprint.

3.7.2 Attività: Monitoraggio continuo

3.7.2.1 Procedure di monitoraggio

Durante lo svolgimento dello sprint, la qualità viene monitorata costantemente attraverso:

1. **Controllo automatizzato:**

- Le GitHub Actions eseguono verifiche automatiche ad ogni commit (indice Gulpease, build success rate)
- La pipeline di CI/CD monitora le metriche di processo come $build_G$ time e test coverage
- Gli strumenti di analisi statica verificano la complessità ciclomatica e altre metriche di codice

2. **Controllo manuale:**
   - I verificatori eseguono revisioni del codice e della documentazione secondo le procedure definite
   - Vengono eseguiti test manuali per verificare usabilità e funzionalità
   - Si monitora l'aderenza alle norme di progetto
3. **Utilizzo del Cruscotto di Qualità:**
   - Le metriche raccolte vengono visualizzate su un cruscotto dedicato all'interno del Piano di Qualifica
   - Vengono monitorati trend e deviazioni rispetto agli obiettivi

3.7.3 Attività: Valutazione periodica

3.7.3.1 Procedure di valutazione

A cadenza regolare vengono effettuate valutazioni approfondite della qualità:

1. **Review settimanale:** Durante le riunioni di team, vengono discusse brevemente le metriche principali e le eventuali criticità emerse
2. **Review di fine sprint:** Al termine di ogni sprint:
   - Si confrontano le metriche ottenute con gli obiettivi pianificati
   - Si analizzano le cause di eventuali scostamenti significativi
   - Si valuta l'efficacia delle azioni correttive implementate
   - Si aggiorna il cruscotto di qualità nel Piano di Qualifica
3. **Review di milestone:** In preparazione delle milestone:
   - Si effettua una valutazione complessiva della qualità raggiunta
   - Si verifica il rispetto di tutti i requisiti di qualità obbligatori

3.7.4 Attività: Azioni correttive

3.7.4.1 Procedure per azioni correttive

Quando il monitoraggio evidenzia deviazioni dagli standard di qualità:

1. **Identificazione della deviazione:** Attraverso il monitoraggio continuo o le review periodiche
2. **Analisi della causa:** Si applica il processo di risoluzione dei problemi (3.6) per identificare le cause radice
3. **Pianificazione dell'intervento:**
    - Per deviazioni minori: si assegna una issue correttiva al membro responsabile
    - Per deviazioni significative: il responsabile convoca una sessione dedicata per pianificare l'intervento
4. **Implementazione e verifica:** L'azione correttiva viene implementata e verificata secondo le normali procedure
5. **Monitoraggio dell'efficacia:** Si verifica che l'azione correttiva abbia effettivamente risolto il problema senza introdurre nuove criticità

3.7.5 Attività: Procedure generali di gestione della qualità

3.7.5.1 Procedure di gestione della qualità

Le seguenti procedure guidano il Team nella gestione sistematica della qualità:

1. **Prevenzione > Correzione:** Si privilegia l'adozione di pratiche preventive (code review, test automatici, standard di codifica) rispetto alla correzione a posteriori
2. **Automazione quando possibile:** Le verifiche ripetitive vengono automatizzate per ridurre errori umani e liberare tempo per attività a maggior valore aggiunto
3. **Tracciabilità completa:** Ogni decisione relativa alla qualità, ogni deviazione e ogni azione correttiva vengono documentate per garantire tracciabilità
4. **Miglioramento continuo:** Le metriche e le procedure di qualità vengono periodicamente riviste e aggiornate in base all'esperienza accumulata
5. **Responsabilità condivisa:** La qualità è responsabilità di tutti i membri del team, non solo dei verificatori

3.7.6 Metriche di gestione della qualità

L'efficacia del processo di gestione della qualità viene valutata attraverso:

- **Percentuale di metriche rispettate:** Numero di metriche che soddisfano le soglie minime rispetto al totale delle metriche monitorate
- **Tasso di ricorrenza delle non conformità:** Percentuale di non conformità che si ripetono dopo essere state risolte
- **Copertura delle verifiche:** Percentuale di deliverable che hanno superato tutte le verifiche previste

Queste meta-metriche vengono rendicontate nel Piano di Qualifica insieme alle metriche di processo e prodotto definite nella sezione 5.

3.7.7 Strumenti di gestione della qualità

- **Piano di Qualifica:** Documento di riferimento per obiettivi, metriche e soglie di qualità
- **Jira e Dashboard Jira (4.4):** Per il monitoraggio in tempo reale dello stato delle metriche, il tracciamento delle non conformità, le azioni correttive e la condivisione di report di qualità con la proponente
- **GitHub Actions (4.4):** Per l'automazione delle verifiche di qualità (Gulpease, build, test)
- **Strumenti di analisi statica:** Per il calcolo automatico di metriche di codice (complessità, coverage, accoppiamento)

4 Processi Organizzativi

4.1 Gestione

La gestione descrive le modalità di comunicazione interne ed esterne, i vari ruoli e i loro compiti.

4.1.1 Ruoli di progetto

Durante lo sprint ogni membro del gruppo assume un singolo ruolo. Il ruolo può variare da sprint a sprint. Di seguito elenchiamo le responsabilità di ogni ruolo:

- Responsabile: gestisce la distribuzione di attività tra i membri, assegnando le attività basandosi sull'ammontare ore rimasto per ogni ruolo ed ogni membro. Si occupa inoltre della redazione dei verbali, della comunicazione con la proponente e dell'aggiornamento del piano di progetto.

- Amministratore: si occupa del corretto svolgimento del progetto, manutenendo l'infrastruttura, e controllando il corretto aderimento alle pratiche sprint. Si occupa anche della scrittura delle norme di progetto.
- Analista: concepisce i casi d'uso basandosi sui requisiti ricavati dalle richieste della proponente. Si occupa della scrittura dell'analisi dei requisiti e del piano di qualifica.
- Progettista: traduce i requisti individuati dagli analisti in un progetto, questo include la scelta delle tecnologie da utilizzare, la struttura interna del progetto, cioè le sue parti e sotto-parti. Si occupa anche della supervisione dello sviluppo, verificando il corretto aderimento al progetto.
- Programmatore: il suo ruolo è sviluppare il progetto generato dal progettista. Lavorando con quest'ultimo implementa tutte le funzionalità richieste tramite le tecnologie prestabilite. Si occupa anche della creazione di test automatizzati, utilizzati per verificare la correttezza del codice sviluppato.
- Verificatore: si occupa di verificare che il prodotto sia a regola d'arte, dunque che aderisca alle norme di progetto. Si occupa di controllare la documentazione e il codice in cerca di possibili errori, che siano stilistici o di contenuto, e avvisare gli autori in questione del problema.

4.1.2 Attività

Appoggiandoci allo standard ISO/IEC 12207:1997 individuiamo le seguenti attività:

- Processo di Gestione.
- Processo di Infrastruttura.
- Processo di Miglioramento.
- Processo di Formazione.

Le sezioni di seguito tratteranno i processi sopra scritti in dettaglio.

4.2 Gestione

4.3 Infrastruttura

Il processo di Infrastruttura fornisce il supporto tecnico necessario per lo sviluppo e la gestione del progetto.

4.3.1 Attività di processo

Il processo si articola nelle seguenti attività principali:

- **Implementazione:** scelta, configurazione e gestione degli strumenti e delle tecnologie necessarie per supportare le attività di progetto.

- **Creazione:** sviluppo e manutenzione dell'infrastruttura tecnica, strumenti, procedure e ambienti di sviluppo.
- **Manutenzione:** aggiornamento, monitoraggio e risoluzione di eventuali problemi legati all'infrastruttura esistente.

4.3.2 Procedure di processo

Le seguenti procedure guidano il Team nell'utilizzo degli strumenti di infrastruttura, garantendo un uso coerente ed efficiente delle risorse disponibili.

1. **Integrazione:** gli strumenti selezionati vengono integrati nell'ambiente di sviluppo esistente.
2. **Documentazione:** viene mantenuta una documentazione aggiornata sugli strumenti utilizzati, le loro configurazioni e le procedure operative. Così facendo tutto il team può accedere facilmente alle informazioni necessarie per utilizzare gli strumenti in modo efficace.

4.3.2.1 Strumenti di Creazione

Gli strumenti adottati per il testing delle tecnologie e la documentazione sono i seguenti.

4.3.2.2 Jira

Come descritto in 3.1.1, è necessaria la creazione di una issue concordata in fase di pianificazione per ogni nuovo strumento da adottare. In questo modo si garantisce la tracciabilità della decisione e la documentazione delle motivazioni alla base della scelta.

4.3.2.3 GitHub

Strumento per l'hosting utilizzato per il testing delle nuove tecnologie. Ogni nuova tecnologia deve essere testata nell'apposita repository o branch, per evitare di compromettere l'infrastruttura esistente. Per gli standard di versionamento e gestione delle branch si rimanda a 3.2.3.1.

4.3.3 Attività: Manutenzione

L'attività di Manutenzione riguarda l'aggiornamento, il monitoraggio e la risoluzione di eventuali problemi legati agli strumenti e alle tecnologie utilizzati nel progetto.

4.3.3.1 Procedure di Manutenzione

Le seguenti procedure guidano il Team nella manutenzione degli strumenti di infrastruttura, garantendo la loro efficienza e affidabilità nel tempo.

1. **Monitoraggio continuo:** viene effettuato un monitoraggio costante delle prestazioni degli strumenti per identificare eventuali problemi o inefficienze.

2. **Aggiornamenti regolari:** gli strumenti vengono aggiornati regolarmente per garantire la sicurezza e l'efficienza.
3. **Supporto tecnico:** viene fornito supporto tecnico ai membri del team per l'utilizzo degli strumenti di infrastruttura.
4. **Disaster Recovery:** viene mantenuto un'analisi dei rischi per i processi di progetto.

In vista dell'attività di produzione di codice, l'infrastruttura verrà aggiornata e ampliata per poter rimanere al passo con le nuove necessità. Sono in particolare previste le seguenti aggiunte:

- **Tracciamento dei requisiti:** Deploy di uno strumento automatico per il tracciamento dei requisiti che ammetta codice solo se i requisiti sono stati tracciati correttamente.
  Tale codice è in fase di scrittura e verrà integrato non appena testato.
- **Integrazioni di Analisi Statica e Dinamica:** Deploy di strumenti di analisi statica e dinamica del codice per garantire la qualità del software prodotto.
- **CI/CD:** Implementazione di pipeline di integrazione continua e distribuzione continua per automatizzare il processo di build, test e rilascio del software.

4.3.3.2 Strumenti di Manutenzione

Gli strumenti adottati per il monitoraggio e la manutenzione dell'infrastruttura sono i seguenti.

- **Jira:** Tracciamento delle attività di manutenzione e monitoraggio degli interventi effettuati.
- **GitHub:** Gestione dei branch di hotfix e documentazione degli update di sicurezza e stabilità.

4.3.3.3 Criteri di Scelta degli Strumenti

La scelta degli strumenti di infrastruttura segue i seguenti criteri misurabili:

- **Costo:** Valutare il rapporto costo-beneficio e la sostenibilità economica a lungo termine.
- **Scalabilità:** Verificare la capacità dello strumento di adattarsi alle crescenti esigenze del progetto.
- **Facilità d'uso:** Assicurare che lo strumento sia intuitivo e richieda un tempo di apprendimento minimo.
- **Supporto e Comunità:** Valutare la disponibilità di documentazione, supporto tecnico e comunità attiva.

- **Integrazione:** Verificare la compatibilità con gli strumenti già in uso nel progetto.
- **Sicurezza:** Assicurare che lo strumento rispetti gli standard di sicurezza e privacy richiesti dal progetto.

4.4 Knowledge Base

In questa sezione sono contenute le informazioni principali sugli strumenti implementati dal gruppo, completi di istruzioni per le singole procedure e best practice adottate.

4.4.0.1 GitHub

Piattaforma di hosting per il versionamento e la gestione dei contenuti di progetto. Il Team deve sfruttare appieno le potenzialità di GitHub, in particolare per l'integrazione con Jira. Vengo quindi descritti gli Smart Commit, lo standard per la scrittura di commit e la gestione dello stato di vita degli work item.

4.4.0.2 Jira

Strumento di gestione delle attività di progetto adottato per potenzialità e flessibilità del sistema. Permette di tracciare le attività di progetto, assegnarle ai membri del gruppo, monitorare lo stato di avanzamento e gestire le scadenze.

4.4.0.2.1 Automation

Il Team ha deciso di adottare alcune automazioni per facilitare la gestione delle attività di progetto. Le automazioni attualmente implementate sono:

- Make child work items inherit labels from parent work items
- When a commit is made then move issue to in progress
- When all child work items are completed then close parent
- When all sub-tasks are done move parent to done
- When Item In Progress Parent In Progress
- Diario Assegnato Creo Preparazione Slide
- Riunione Assegnata Creo Scrittura Verbale

4.4.0.3 VSCode

Ambiente di sviluppo integrato (IDE) utilizzato per la scrittura del codice e la gestione della documentazione di progetto. Grazie all'estensione Atlassian per **Jira** e **GitHub**, il Team può integrare direttamente le funzionalità di gestione delle attività e del versionamento all'interno dell'IDE, migliorando l'efficienza del flusso di lavoro e uniformando le pratiche di sviluppo.
Si presuppone che tutti i membri del gruppo si adattino allo standard comune. Le estensioni attualmente utilizzate sono:

- **Atlassian: Jira, Rovo Dev, Bitbucket**
- **GitHub Pull Requests and Issues**
- **GitHub Actions**
- **GitHub Codespaces**
- **LaTeX Workshop**

Altre estensioni come GitHub Copilot possono essere utilizzate a discrezione del membro del gruppo, al fine di velocizzare, per esempio, il processo di Documentazione 3.1.

4.4.0.3.1 Atlassian: Jira, Rovo Dev, Bitbucket

Estensione per l'integrazione di Jira e GitHub in VSCode. Permette di visualizzare e gestire le issue Jira direttamente dall'IDE, creare branch di lavoro basati sulle issue e monitorare lo stato di avanzamento delle attività.

4.4.0.3.2 GitHub Pull Requests and Issues

Estensione per la gestione delle pull request e delle issue GitHub. Permette di creare, revisionare e gestire le pull request direttamente dall'IDE, migliorando l'efficienza del processo di revisione del codice.

4.4.0.4 Google Presentazioni

Strumento utilizzato per la creazione dei Diari di Bordo. La scelta di questo strumento è dovuta alla sua facilità d'uso e alla possibilità di collaborare in tempo reale tra i membri del gruppo. Cosi è possibile tracciare le problematiche e i dubbi emersi durante lo svolgimento delle attività di progetto. Per i DIari di Bordo è stato impostato un template presente nel Google Drive di gruppo.

4.4.0.5 Google Drive

Strumento di archiviazione e condivisione dei file con l'azienda proponente e per la condivisione rapida dei Diari di Bordo.
Questo strumento è inoltre utilizzato come mezzo di condivisione dei file con **Sanmarco Informatica**, quali materiali formativi, materiale d'esempio e normative aziendali. Sempre tramite Google Drive vengono condivisi i verbali esterni con l'azienda proponente, in modo da poter essere approvati e archiviati in modo sicuro.

4.4.0.6 Google Mail

Strumento di comunicazione formale con l'azienda proponente e per la gestione delle comunicazioni ufficiali del gruppo.
Tutti i membri del gruppo devono utilizzare l'account di posta elettronica ufficiale del gruppo per le comunicazioni formali, garantendo così la tracciabilità e la professionalità nelle interazioni esterne.
Le comunicazioni esterne sono gestite dal Responsabile salvo diverse indicazioni, il quale deve ricordare che parla a nome di tutto il gruppo.

4.4.0.7 Discord

Strumento di comunicazione principale del gruppo, utilizzato per la coordinazione delle attività, la condivisione di informazioni e la collaborazione tra i membri del team.
L'accesso al server Discord del gruppo è obbligatorio per tutti i membri, in quanto rappresenta il canale ufficiale di comunicazione e supporto.

4.4.0.8 Whatsapp

Strumento di comunicazione informale e sincrono tra i membri del gruppo, utilizzato per la coordinazione rapida delle attività e la condivisione di informazioni urgenti.
Si raccomanda di essere particolarmente responsivi su questo canale, in quanto spesso viene utilizzato per comunicazioni che richiedono una risposta tempestiva.

4.5 Processo di Miglioramento

Il processo di miglioramento continuo mira a identificare e implementare modifiche ai processi e alle pratiche di progetto per aumentare l'efficienza e la qualità del lavoro svolto.

4.5.1 Attività: Analisi degli Incidenti

L'attività di Analisi degli Incidenti riguarda l'identificazione, documentazione e risoluzione dei problemi riscontrati durante l'utilizzo degli strumenti e dei processi di progetto.

4.5.1.1 Procedure di Analisi degli Incidenti

Le seguenti procedure guidano il Team nella gestione degli incidenti e nella formulazione di soluzioni:

1. **Segnalazione:** Qualsiasi membro del team può segnalare un incidente tramite Jira creando un'issue con label "incident".
2. **Analisi della causa radice:** Il Responsabile coordina l'analisi per identificare le cause sottostanti.
3. **Implementazione della soluzione:** Una volta identificata la causa, viene pianificata e implementata una soluzione.
4. **Verifica della risoluzione:** Viene verificato che la soluzione risolva effettivamente l'incidente.
5. **Archiviazione e apprendimento:** L'incidente e la soluzione vengono archiviati per consultazione futura e per evitare ricorrenze. Tali informazioni vengono tracciate nei Verbali Interni e nei Piano di Progetto.

4.5.2 Attività: Monitoraggio degli Strumenti

Il monitoraggio degli strumenti è un'attività fondamentale per garantire che tutti i membri del gruppo utilizzino gli strumenti in modo efficace e coerente. Questo include il controllo dell'utilizzo dei tool di comunicazione, collaborazione e gestione del progetto, nonché l'identificazione di eventuali problemi o inefficienze.

4.5.2.1 Procedure di Monitoraggio

Le seguenti procedure guidano il Team nel monitoraggio degli strumenti di progetto, garantendo un uso coerente ed efficiente delle risorse disponibili.

1. **Raccolta feedback:** viene raccolto il feedback dai membri del team sull'efficacia degli strumenti utilizzati.
2. **Analisi delle prestazioni:** vengono analizzate le prestazioni degli strumenti per identificare eventuali problemi o inefficienze. A questo scopo si possono utilizzare metriche specifiche per lo strumento e la task presa in considerazione.

4.5.2.2 Strumenti di Monitoraggio

Gli strumenti adottati per il monitoraggio degli strumenti di progetto sono i seguenti.

- **Jira:** Utilizzato per tracciare le attività di monitoraggio e raccogliere feedback dai membri del team.
- **Discord:** Utilizzato per comunicare con i membri del team e raccogliere feedback sugli strumenti utilizzati.
- **Whatsapp:** Utilizzato per comunicazioni rapide e raccolta di feedback urgenti.

4.5.3 Attività: Modifiche all'Infrastruttura

L'attività di Modifiche all'Infrastruttura riguarda l'implementazione di modifiche agli strumenti e alle tecnologie utilizzate nel progetto, al fine di migliorarne l'efficienza e l'affidabilità.

4.5.3.1 Procedure di Modifica

Le seguenti procedure guidano il Team nell'implementazione delle modifiche all'infrastruttura di progetto, garantendo un uso coerente ed efficiente delle risorse disponibili.

1. **Pianificazione delle modifiche:** viene pianificata l'implementazione delle modifiche, definendo gli obiettivi, le risorse necessarie e i tempi di esecuzione.
2. **Implementazione delle modifiche:** le modifiche vengono implementate secondo la pianificazione stabilita e seguendo le stesse norme descritte in **??** e **??**.
3. **Verifica delle modifiche:** viene effettuata una verifica delle modifiche per garantire che siano state implementate correttamente e che abbiano raggiunto gli obiettivi prefissati.

4.5.3.2 Strumenti di Modifica

Gli strumenti adottati per l'implementazione delle modifiche all'infrastruttura di progetto sono i seguenti.

- **Jira:** Utilizzato per tracciare le attività di modifica e monitorare lo stato di avanzamento.
- **GitHub:** Utilizzato per gestire i branch o repository di testing degli strumenti e per la documentazione di questi ultimi.

4.5.4 Frequenza dei Cicli di Miglioramento

I cicli di miglioramento continuo seguono una cadenza regolare:

- **Review settimanale:** Durante le riunioni di team, vengono discussi brevemente i feedback e i problemi riscontrati.
- **Review mensile:** Una sessione dedicata mira a identificare modelli nei feedback e proporre miglioramenti significativi.
- **Review trimestrale:** Una valutazione più approfondita della qualità complessiva dei processi e dell'infrastruttura.

4.5.5 Metriche di Miglioramento

Il progresso dei miglioramenti implementati viene misurato attraverso le seguenti metriche:

- **Tempo di Risoluzione degli Incidenti:** Media del tempo impiegato per risolvere i problemi segnalati (target: < 48 ore per problemi critici).
- **Satisfaction Index:** Feedback qualitativo raccolto dai membri del team sulla soddisfazione riguardo gli strumenti e i processi.
- **Adoption Rate:** Percentuale di utilizzo effettivo dei nuovi strumenti o processi implementati.
- **Numero di Incidenti Ricorrenti:** Monitoraggio dei problemi che si ripetono, indicatore di efficacia della risoluzione.

4.6 Processo di Formazione

Il processo di formazione mira a garantire che tutti i membri del gruppo abbiano le competenze e le conoscenze necessarie per svolgere efficacemente le loro attività di progetto.

4.6.1 Attività: Pianificazione della Formazione

La pianificazione della formazione è un'attività fondamentale per garantire un'efficace trasmissione delle conoscenze. La pianificazione di queste attività segue la pianificazione generale del progetto, in modo da integrare la formazione con le altre attività di progetto.

4.6.1.1 Procedure di Pianificazione

Le seguenti procedure guidano il Team nella pianificazione delle attività di formazione, garantendo un uso coerente ed efficiente delle risorse disponibili.

1. **Identificazione delle necessità formative:** vengono identificate le necessità formative dei membri del team, in base alle competenze richieste per le attività di progetto.
2. **Pianificazione delle attività formative:** vengono pianificate le attività formative, definendo gli obiettivi, i contenuti, le risorse necessarie e i tempi di erogazione.
3. **Allocazione delle risorse:** vengono allocate le risorse umane e strumentali necessarie per garantire l'efficacia della formazione.

4.6.2 Attività: Erogazione della Formazione

L'erogazione della formazione si implementa attraverso sessioni di formazione. In particolare, sono identificabili due tipologie di sessioni:

4.6.2.1 Procedure di Erogazione

Le seguenti procedure guidano il Team nell'erogazione delle attività di formazione e la condivisione delle conoscenze al fine di migliorare le competenze dell'intero gruppo.

- **Formazione Autonoma:** Sessioni di formazione condotte dai membri del gruppo con competenze specifiche su determinati argomenti e strumenti in modo individuale.
- **Confronto:** Sessioni di formazione condotte da tutti i membri coinvolti su un argomento specifico, con l'obiettivo di condividere conoscenze, best practice e formulare soluzioni comuni.
- **Archiviazione:** I materiali e le note della formazione vengono salvati nelle Norme di Progetto per rendere disponibili a tutti i materiali di Formazione.

4.6.3 Attività: Valutazione della Formazione

L'attività di Valutazione della Formazione serve a verificare l'efficacia delle sessioni di formazione e l'acquisizione delle competenze da parte dei partecipanti.

4.6.3.1 Procedure di Valutazione

Le seguenti procedure guidano il Team nella valutazione dell'efficacia della formazione:

1. **Feedback immediato:** Al termine di ogni sessione di formazione, viene raccolto feedback qualitativo dai partecipanti sulla chiarezza e l'utilità dei contenuti.
2. **Verifica pratica:** Nei giorni successivi alla formazione, vengono assegnati esercizi pratici o task che richiedono l'applicazione delle conoscenze apprese.
3. **Valutazione del risultato:** Le performance nei task assegnati vengono valutate per determinare il livello di acquisizione delle competenze.
4. **Seguito:** Nel caso in cui i risultati della valutazione indichino lacune, viene organizzata formazione integrativa.

4.6.3.2 Strumenti di Valutazione

Gli strumenti utilizzati per la valutazione della formazione includono:

- **Jira:** Per tracciare i task di verifica assegnati post-formazione.
- **Discord:** Per la discussione e il confronto tra i membri coinvolti nella formazione.
- **VSCode:** IDE principale del gruppo che grazie alle integrazioni con Jira e GitHub permette una gestione semplice della procedura di Verifica pratica.

5 Metriche della qualità

Lo standard ISO/IEC 9126-1:1999 individua tre categorie fondamentali di requisiti per la qualità che concorrono alla creazione del prodotto e interconnesse tra loro secondo il Modello a V$_{(Figura\ 5)}$ (*V-Model*$_G$):

- **Bisogni utente**: sono i requisiti che il prodotto deve soddisfare per incontrare i bisogni dell'utente. L'output corrispondente e' la qualità in uso, verificata da apposite metriche esterne.
- **Requisiti di qualità esterni**: corrispondono ai requisiti che riguardano le caratteristiche proprie del prodotto da realizzare. Il loro soddisfacimento definisce la qualità esterna del prodotto, validata anch'essa da opportune metriche esterne.
- **Requisiti di qualità interni**: sono i requisiti legati strettamente ai processi di sviluppo software che quando rispettati risultano in codice di qualità, processi efficaci e organizzazione efficiente.

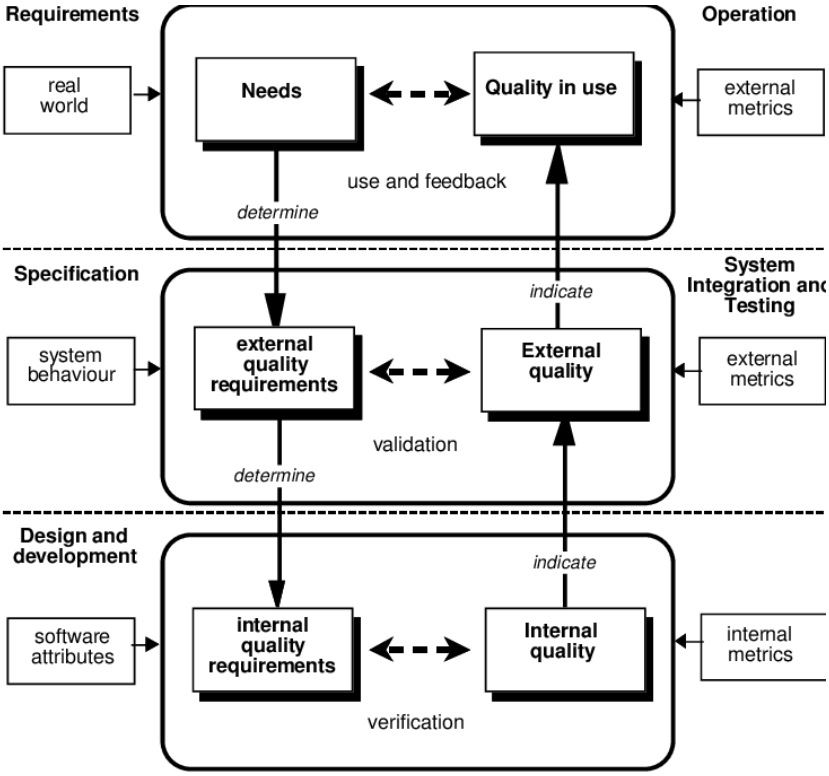

Figura 5: Modello a V

Di seguito vengono indicate le metriche utilizzate per valutare la qualità del prodotto e dei processi di sviluppo.

5.1 Qualità di Processo

In questa sezione vengono definite le metriche utilizzate per valutare l'efficienza e l'efficacia dei processi del ciclo di vita del software, in conformità con lo standard

ISO/IEC 12207.

5.1.1 Processi primari

5.1.1.1 Processo di fornitura

MPC-1 - Earned Value (EV)

Descrizione: Valore del lavoro effettivamente completato in un determinato momento, espresso in termini di budget approvato.

Formula di calcolo:

$$EV = \sum_i (\%\text{Completamento}_i \times \text{Budget}_i)$$

MPC-2 - Planned Value (PV)

Descrizione: Valore del lavoro che si era pianificato di completare entro una certa data.

Formula di calcolo:

$$PV = \sum_i (\%\text{Pianificato}_i \times \text{Budget}_i)$$

MPC-3 - Actual Cost (AC)

Descrizione: Costo realmente sostenuto per il lavoro svolto fino alla data corrente.

Formula di calcolo:

$$AC = \sum \text{Costi Sostenuti}$$

MPC-4 - Cost Performance Index (CPI)

Descrizione: Indice di efficienza dei costi. Un valore < 1 indica che il progetto è fuori budget.

Formula di calcolo:

$$CPI = \frac{EV}{AC}$$

MPC-5 - Schedule Performance Index (SPI)

Descrizione: Indice di efficienza della *schedulazione*$_G$. Un valore < 1 indica che il progetto è in ritardo.

Formula di calcolo:

$$SPI = \frac{EV}{PV}$$

MPC-6 - Estimate to Complete

Descrizione: Rappresenta l'ammontare rimanente stimato per completare il progetto

Formula di calcolo:

$$ETC = EAC - AC$$

MPC-7 - Estimate at Completion (EAC)

Descrizione: Stima totale dei costi alla fine del progetto, basata sulle performance attuali.

Formula di calcolo:

$$EAC = \frac{BAC}{CPI}$$

MPC-6 - Estimate to Complete

Descrizione: Rappresenta l'ammontare rimanente stimato per completare il progetto

Formula di calcolo:

$$ETC = EAC - AC$$

MPC-7 - Estimate at Completion (EAC)

Descrizione: Stima totale dei costi alla fine del progetto, basata sulle performance attuali.

Formula di calcolo:

$$EAC = \frac{BAC}{CPI}$$

5.1.1.2 Processo di Sviluppo

MPC-8 - Deployment Frequency

Descrizione: Frequenza con cui il team rilascia nuove versioni del software, potenzialmente trasportabili in produzione.

Formula di calcolo:

$$DF = \frac{\text{Numero di deployment}}{\text{Periodo di tempo}}$$

MPC-8 - Deployment Frequency

Descrizione: Frequenza con cui il team rilascia nuove versioni del software, potenzialmente trasportabili in produzione.

Formula di calcolo:

$$DF = \frac{\text{Numero di deployment}}{\text{Periodo di tempo}}$$

MPC-9 - Requirements Stability Index

Descrizione: Rappresenta la stabilità dei requisiti del prodotto software nel tempo.

Formula di calcolo:

$$RSI = \frac{\#\text{Req originali} - (\#\text{Req modificati} + \#\text{Req eliminati} + \#\text{Req aggiunti})}{\#\text{Req originali}} \cdot 100$$

5.1.1.3 Processo di Integrazione

MPC-12 - Average Build Time

Descrizione: Tempo medio impiegato per completare il processo di build e test automatici.

Formula di calcolo:

$$T_{avg} = \frac{\sum_{i=1}^{n} T_{build\_i}}{n}$$

MPC-15 - Correttezza Ortografica

Descrizione: Misura la densità di errori ortografici presenti nella documentazione.

Formula di calcolo:

$$C = 1 - \left(\frac{\text{Numero errori ortografici}}{\text{Numero parole totali}}\right)$$

5.1.1.4 Processo di Documentazione

MPC-10 - Indice di Gulpease

Descrizione: Indice che valuta la leggibilità di un testo in lingua italiana. Valori bassi indicano bassa leggibilità.

Formula di calcolo:

$$G = 89 + \frac{300 \times (\#\text{Frasi}) - 10 \times (\#\text{Lettere})}{\#\text{Parole}}$$

MPC-11 - Correttezza Ortografica

Descrizione: Misura la densità di errori ortografici presenti nella documentazione.

Formula di calcolo:

$$C = \#\text{errori ortografici}$$

5.1.1.5 Processo di Verifica

MPC-13 - Code review turnaround time

Descrizione: Tempo medio impiegato per completare una revisione del codice in vista di una Pull Request.

Formula di calcolo:

$$CRT = \frac{\sum_{i=1}^{n} T_{review\_i}}{n}$$

MPC-13 - Code review turnaround time

Descrizione: Tempo medio impiegato per completare una revisione del codice in vista di una Pull Request.

Formula di calcolo:

$$CRT = \frac{\sum_{i=1}^{n} T_{review\_i}}{n}$$

MPC-14 - Test success rate

Descrizione: Percentuale di test superati rispetto al totale dei test eseguiti.

Formula di calcolo:

$$TSR = \left(\frac{\#\text{Test superati}}{\#\text{Test eseguiti}}\right) \times 100$$

MPC-14 - Test success rate

Descrizione: Percentuale di test superati rispetto al totale dei test eseguiti.

Formula di calcolo:

$$TSR = \left(\frac{\#\text{Test superati}}{\#\text{Test eseguiti}}\right) \times 100$$

5.1.2 Processi Organizzativi

5.1.2.1 Processo di Gestione dei Rischi

MPC-15 - Rischi non previsti

Descrizione: Numero di rischi non previsti ad inizio periodo, per ciascun periodo di progetto.

Formula di calcolo:

$$RNP = \#\text{Rischi non previsti}$$

5.1.2.2 Processo di Gestione della qualità

MPC-16 - Metriche soddisfatte

Descrizione: Percentuale di metriche di qualità che vengono soddisfatte in un determinato istante di tempo.

Formula di calcolo:

$$MS = \left(\frac{\#\text{Metriche soddisfatte}}{\#\text{Metriche totali}} \right) \times 100$$

5.2 Qualità di Prodotto

In questa sezione vengono definite le metriche per valutare la qualità intrinseca del prodotto software, basandosi sul modello ISO/IEC 9126.

5.2.0.1 Funzionalità

MPD-1 - Requisiti obbligatori soddisfatti

Descrizione: Percentuale di requisiti funzionali obbligatori implementati e testati.

Formula di calcolo:

$$R_{\text{ob}} = \left(\frac{\text{Requisiti obbligatori soddisfatti}}{\text{Requisiti obbligatori totali}} \right) \times 100$$

MPD-2 - Requisiti opzionali soddisfatti

Descrizione: Percentuale di requisiti funzionali opzionali implementati e testati.

Formula di calcolo:

$$R_{\text{op}} = \left(\frac{\text{Requisiti opzionali soddisfatti}}{\text{Requisiti opzionali totali}} \right) \times 100$$

MPD-1 - Requisiti obbligatori soddisfatti

Descrizione: Percentuale di requisiti funzionali obbligatori implementati e testati.

Formula di calcolo:

$$R_{\text{ob}} = \left(\frac{\text{Requisiti obbligatori soddisfatti}}{\text{Requisiti obbligatori totali}} \right) \times 100$$

MPD-2 - Requisiti opzionali soddisfatti

Descrizione: Percentuale di requisiti funzionali opzionali implementati e testati.

Formula di calcolo:

$$R_{\text{op}} = \left(\frac{\text{Requisiti opzionali soddisfatti}}{\text{Requisiti opzionali totali}} \right) \times 100$$

MPD-3 - Requisiti desiderabili soddisfatti

Descrizione: Percentuale di requisiti funzionali desiderabili implementati e testati.

Formula di calcolo:

$$R_{\text{de}} = \left(\frac{\text{Requisiti desiderabili soddisfatti}}{\text{Requisiti desiderabili totali}} \right) \times 100$$

MPD-3 - Requisiti desiderabili soddisfatti

Descrizione: Percentuale di requisiti funzionali desiderabili implementati e testati.

Formula di calcolo:

$$R_{\text{de}} = \left(\frac{\text{Requisiti desiderabili soddisfatti}}{\text{Requisiti desiderabili totali}} \right) \times 100$$

5.2.0.2 Affidabilità

MPD-5 - Broken links

Descrizione: Numero di riferimenti ipertestuali all'interno dell'applicazione che non portano alla risorsa corretta o non portano ad alcuna risorsa.

Formula di calcolo:

$$BL = \#\text{Broken Links}$$

MPD-6 - Statement Coverage

Descrizione: Percentuale di istruzioni (statements) del codice sorgente eseguite durante i test automatici.

Formula di calcolo:

$$SC = \left(\frac{\text{Linee di codice eseguite}}{\text{Linee di codice totali}} \right) \times 100$$

MPD-5 - Broken links

Descrizione: Numero di riferimenti ipertestuali all'interno dell'applicazione che non portano alla risorsa corretta o non portano ad alcuna risorsa.

Formula di calcolo:

$$BL = \#\text{Broken Links}$$

MPD-6 - Statement Coverage

Descrizione: Percentuale di istruzioni (statements) del codice sorgente eseguite durante i test automatici.

Formula di calcolo:

$$SC = \left(\frac{\text{Linee di codice eseguite}}{\text{Linee di codice totali}} \right) \times 100$$

MPD-7 - Branch Coverage

Descrizione: Percentuale di rami decisionali (if, switch, loop) percorsi durante i test.

Formula di calcolo:

$$BC = \left(\frac{\text{Rami percorsi}}{\text{Rami totali}} \right) \times 100$$

MPD-7 - Branch Coverage

Descrizione: Percentuale di rami decisionali (if, switch, loop) percorsi durante i test.

Formula di calcolo:

$$BC = \left(\frac{\text{Rami percorsi}}{\text{Rami totali}} \right) \times 100$$

5.2.0.3 Usabilità

MPD-8 - Profondità di Navigazione

Descrizione: Numero massimo di click necessari per raggiungere una qualsiasi pagina di contenuto partendo dalla Home Page.

Formula di calcolo:

$$\sum_{i=1}^{n} \frac{\text{Click}_i}{n} \quad \begin{array}{l} (\text{dove } n = \text{numero di pagine}) \\ (\text{dove Click}_i = \text{numero di click per raggiungere la pagina } i) \end{array}$$

MPD-8 - Profondità di Navigazione

Descrizione: Numero massimo di click necessari per raggiungere una qualsiasi pagina di contenuto partendo dalla Home Page.

Formula di calcolo:

$$\sum_{i=1}^{n} \frac{\text{Click}_i}{n} \quad \begin{array}{l} (\text{dove } n = \text{numero di pagine}) \\ (\text{dove Click}_i = \text{numero di click per raggiungere la pagina } i) \end{array}$$

5.2.0.4 Efficienza

MPD-9 - Indexing time

Descrizione: Tempo di *indicizzazione*$_G$ per un DiP di dimensioni standard (1-4GB).

Formula di calcolo:

$$IT = \text{Tempo di indicizzazione (secondi)}$$

MPD-10 - Search Time

Descrizione: Tempo medio di risposta per un set di query predefinite.

Formula di calcolo:

$$ST = \sum_{i=1}^{n} \frac{\text{Tempo di risposta}_i}{n} \quad \begin{matrix}(\text{dove } n = \text{numero di query}) \\ (\text{dove } i = \text{indice della query})\end{matrix}$$

MPD-11 - Average CPU usage

Descrizione: Percentuale media di utilizzo della CPU durante l'esecuzione del software. Si considera un campione di misurazioni effettuate in condizioni di carico (durante l'indicizzazione o la ricerca) nonché in idle. Quest'ultimo per assicurare che il software non consumi risorse inutilmente quando non è in uso.

Formula di calcolo:

$$CPU_{avg} = \sum_{i=1}^{n} \frac{\text{Utilizzo CPU}_i}{n} \quad (\text{dove } n = \text{numero di misurazioni})$$

MPD-12 - Peak memory usage

Descrizione: Massima quantità di memoria RAM utilizzata durante l'esecuzione del software. Permette di individuare eventuali memory leak o scarsa gestione delle risorse.

Formula di calcolo:

$$MEM_{peak} = \max(\text{Utilizzo RAM}_i) \quad (\text{dove } i = 1, \ldots, n, \text{istante di tempo } i - \text{esimo})$$

5.2.0.5 Manutenibilità

MPD-13 - Complessità Ciclomatica

Descrizione: Misura la complessità del flusso di controllo del codice. Valori superiori a 15 indicano codice difficile da testare e manutenere.

MPD-13 - Complessità Ciclomatica

Descrizione: Misura la complessità del flusso di controllo del codice. Valori superiori a 15 indicano codice difficile da testare e manutenere.

Formula di calcolo:

$$M = E - N + 2P \quad (\text{dove } E = \text{archi}, N = \text{nodi}, P = \text{componenti connessi})$$

MPD-14 - Accoppiamento tra Classi (CBO)

Descrizione: Numero di classi a cui una determinata classe è accoppiata (usa o è usata da). Un valore alto indica scarsa modularità.

Formula di calcolo:

$$CBO = \frac{\#\text{Dipendenze}}{\#\text{Componenti}}$$

MPD-14 - Accoppiamento tra Classi (CBO)

Descrizione: Numero di classi a cui una determinata classe è accoppiata (usa o è usata da). Un valore alto indica scarsa modularità.

Formula di calcolo:

$$CBO = \frac{\#\text{Dipendenze}}{\#\text{Componenti}}$$

MPD-15 - Code smells

Descrizione: Numero di code smells, misurato ogni 1000 righe di codice (KLOC). I code smells sono indicatori di potenziali problemi di design o implementazione.

MPD-15 - Code smells

Descrizione: Numero di code smells, misurato ogni 1000 righe di codice (KLOC). I code smells sono indicatori di potenziali problemi di design o implementazione.

5.2.0.6 Portabilità

MPD-16 - SO Compatibility

Descrizione: Numero di sistemi operativi supportati dal software. Un valore alto indica una maggiore portabilità.

Formula di calcolo:

$$SO_{comp} = \#\text{Sistemi operativi supportati}$$

5.2.0.7 Portabilità

MPD-16 - SO Compatibility

Descrizione: Numero di sistemi operativi supportati dal software. Un valore alto indica una maggiore portabilità.

Formula di calcolo:

$$SO_{comp} = \#\text{Sistemi operativi supportati}$$

5.3 Strumenti

5.3.0.1 Jira Software

Utilizzato come strumento di Project Management per il tracciamento dei requisiti e la gestione del workflow.

- **Metriche associate:** MPC-1 ...MPC-7 (Earned Value Management), MPC-16 (Metriche soddisfatte), MPD-1 ...MPD-3 (Requisiti).
- **Utilizzo:** Attraverso la funzione di esportazione $JQL_G \rightarrow$ csv, e uno script python appositamente sviluppato, vengono estratti i dati necessari per il calcolo delle metriche di Earned Value e dei requisiti soddisfatti, permettendo un monitoraggio costante dell'andamento del progetto.

5.3.0.2 GitHub & GitHub Actions

Viene utilizzato per il controllo di versione e l'automazione della CI/CD.

- **Metriche associate:** MPC-8 (Deployment Frequency), MPC-12 (Build Time), MPC-13 (Code Review Turnaround Time), MPC-14 (Test success rate).

- **Utilizzo:** I workflow di GitHub Actions oltre che garantire la metrica MPC-14 (Test success rate) attraverso l'esecuzione di test automatici, permettono di tracciare i tempi di build e code review, fornendo dati utili per il calcolo delle metriche di processo.

5.3.1 Strumenti di Analisi Statica e Qualità del Codice

5.3.1.1 SonarQube

Piattaforma centrale per la *Continuous Inspection* del codice TypeScript (Angular) e JavaScript (Electron).

- **Metriche associate:** MPD-6 (Statement Coverage), MPD-7 (Branch Coverage), MPD-13 (Complessità Ciclomatica), MPD-14 (CBO), MPD-15 (Code Smells).
- **Utilizzo:** Analizza staticamente il codice ad ogni *Push*$_G$, verificando che il superamento dei *Quality Gates*$_G$ stabiliti sia coerente con le soglie definite nelle metriche di manutenibilità.

5.3.1.2 Aspell & Script Python (Custom)

Data la natura della documentazione in lingua italiana, si rende necessario uno strumento specifico per l'analisi testuale.

- **Metriche associate:** MPC-10 (Indice di Gulpease), MPC-11 (Correttezza Ortografica).
- **Utilizzo:** Uno script Python che utilizza la libreria `textstat` per il calcolo dell'indice di Gulpease e le API di *Aspell* per il conteggio degli errori ortografici nei file Markdown/LaTeX.

5.3.2 Strumenti di Testing e Performance (Electron-Angular)

5.3.2.1 Cypress

Framework di testing End-to-End utilizzato per simulare l'interazione utente all'interno dell'ambiente Electron.

- **Metriche associate:** MPD-8 (Profondità di Navigazione).
- **Utilizzo:** Viene configurato un *crawler* automatizzato che percorre l'albero di navigazione dell'applicazione Angular, validando la raggiungibilità dei link e calcolando il numero di interazioni (click) necessarie per raggiungere i nodi foglia.

5.3.2.2 Electron Manager & Chrome DevTools Protocol

Per misurare l'efficienza di un'applicazione Electron, è necessario monitorare l'istanza di Chromium sottostante.

- **Metriche associate:** MPD-11 (Average CPU usage), MPD-12 (Peak memory usage).
- **Utilizzo:** Durante le sessioni di test automatizzati con Cypress, vengono invocati script Node.js che interrogano le API `process.getProcessMemoryInfo()` e `process.getCPUUsage()` di Electron per campionare il consumo di risorse in idle e sotto carico.

5.3.2.3 TotalValidator

Integrato nei tool di sviluppo, viene utilizzato per profilare i tempi di risposta dell'interfaccia Angular.

- **Metriche associate:** MPD-5 (Broken links).
- **Utilizzo:** Controllo formale di consistenza e correttezza della struttura dell'interfaccia utente Angular.

5.3.2.4 Funzioni interne di misurazione performance

Integrate direttamente nel codice sorgente, queste funzioni permettono di misurare i tempi di indicizzazione e risposta alle query, fornendo dati precisi per le metriche MPD-9 (Indexing time) e MPD-10 (Search Time).