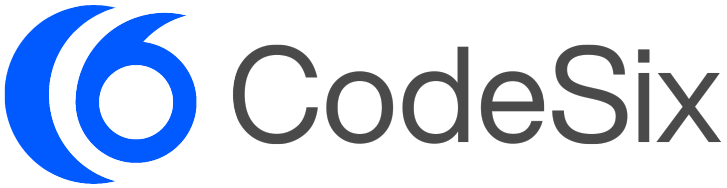


Norme di Progetto

Progetto Ingegneria Del Software

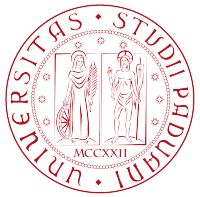
Versione: 1.0.0

Albertin Enrico
Davide Spada
Bettin Michele

Marcatti Pietro
Marco Andrea Limongelli
Matteo Raccanello

Dipartimento di Matematica
Università degli Studi di Padova

March 1, 2022



Registro delle Modifiche

Versione	Modifica	Ruolo	Esecutore	Data
1.0.0	Approvazione del documento	Responsabile	Marcatti Pietro	11/01/22
0.5.0	Verifica complessiva	Verificatore	Limongelli Marco Andrea	10/01/22
0.4.2	Stesura sezione Formazione 4.3 e verifica	Analista, Verificatore	Marcatti Pietro, Davide Spada	09/01/21
0.4.1	Stesura sezione e Codifica 2.2.4.3 e verifica	Analista, Verificatore	Limongelli Marco Andrea, Bettin Michele	07/01/21
0.4.0	Verifica complessiva	Verificatore	Raccanello Matteo	05/01/22
0.3.3	Stesura sezione Gestione infrastrutture 4.2 e verifica	Analista, Verificatore	Marcatti Pietro, Davide Spada	05/01/22
0.3.2	Stesura sezione Progettazione 2.2.4.2 e verifica	Analista, Verificatore	Marcatti Pietro, Davide Spada	03/01/22
0.3.1	Stesura sezione Validazione 3.5 e verifica	Analista, Verificatore	Bettin Michele, Marcatti Pietro	31/12/21
0.3.0	Verifica complessiva	Verificatore	Raccanello Matteo	30/12/21
0.2.2	Stesura sezione Gestione della qualità 3.3 e verifica	Analista, Verificatore	Davide Spada, Bettin Michele	27/12/21
0.2.1	Stesura sezione Gestione della configurazione 3.2 e verifica	Analista, Verificatore	Bettin Michele, Davide Spada	26/12/21
0.2.0	Verifica complessiva	Verificatore	Enrico Albertin	23/12/21
0.1.3	Stesura sezione Metriche 2.2.5 e verifica	Analista, Verificatore	Limongelli Marco Andrea, Spada Davide	23/12/21
0.1.2	Stesura sezione Sviluppo 2.2 [2.2.1 - 2.2.4.1] e verifica	Analista, Verificatore	Spada Davide, Limongelli Marco Andrea	22/12/21
0.1.1	Stesura sezione Organizzazione interna 4.1 e verifica	Analista, Verificatore	Marcatti Pietro, Bettin Michele	20/12/21
0.1.0	Verifica complessiva	Verificatore	Limongelli Marco Andrea	19/12/21

0.0.4	Stesura sezione Verifica 3.4 e verifica	Analista, Verificatore	Spada Davide, Enrico Albertin	18/12/21
0.0.3	Stesura sezione Fornitura 2.1 e verifica	Analista, Verificatore	Enrico Albertin, Marcatti Pietro	14/12/21
0.0.2	Stesura sezione Introduzione 1 e Documentazione 3.1 e verifica	Analista, Verificatore	Enrico Albertin, Marcatti Pietro	12/12/21
0.0.1	Stesura iniziale dello scheletro del documento e verifica	Analista, Verificatore	Marcatti Pietro, Enrico Albertin	10/12/21

Contents

1 Introduzione

1.1 Scopo del documento

Questo documento ha lo scopo di definire le linee guida e il way of working* che si applicano a tutti i processi istanziati dal gruppo CodeSix. Ogni membro del team è tenuto a rispettare quanto scritto in questo documento al fine di assicurare coerenza e uniformità. Vengono normate l'organizzazione, le convenzioni abbracciate dal team nell'uso delle tecnologie, nella scrittura di codice e nella redazione.

1.2 Scopo del capitolato

Il capitolato C5 proposto da Zucchetti ha lo scopo di illustrare l'idea di un prodotto software che ci impegniamo a realizzare, secondo le specifiche dettate nel documento di Analisi dei Requisiti. Si vuole creare una web app* che sfrutti la visualizzazione grafica per effettuare la fase di analisi esplorativa delle informazioni raccolte dai tentativi di login ai server aziendali. Per permettere di gestire anche grandi quantità di dati su molte dimensioni si fa uso di algoritmi per la riduzione dimensionale che alleggerisce il carico computazionale e aiuta a evidenziare trend* nei dati.

1.3 Glossario

Al fine di minimizzare le ambiguità il team ha prodotto ***Glossario 1.0.0*** che raccoglie termini di particolare importanza o ai quali è assegnato un significato particolare.

1.4 Riferimenti

1.4.1 Riferimenti normativi

- **Capitolato di appalto C5 - Login Warrior**

1.4.2 Riferimenti informativi

- Software Engineering - Ian Sommerville: 10th Edition
  - Capitolo 25 - Configuration management
- **Standard ISO/IEC 12207:1995**
- **La qualità del Software secondo il modello ISO/IEC 9126 - Ercole F. Colonese**
  - Capitolo 2 - Il modello ISO/IEC 9126;
  - Capitolo 3 - Le metriche della qualità del software;
  - Capitolo 4
    - Capitolo 4.4 - Esempio di metriche interne
    - Capitolo 4.6 - Esempio di metriche esterne
- **Processi di Vita - Materiale didattico del corso di Ingegneria del Software**
  - Standard di Processo, Slide 8;
  - ISO/IEC 12207:1995 Processi primari, di supporto e organizzativi, Slide 10-14;
  - Alcune attività di processo, Slide 17.
- **Amministrazione di progetto - Materiale didattico del corso di Ingegneria del Software**
  - Normare il *way of working**, Slide 3;
  - Ambiente di lavoro, Supporto a sviluppo, Slide 4-7.

2 Processi primari

2.1 Fornitura

2.1.1 Scopo

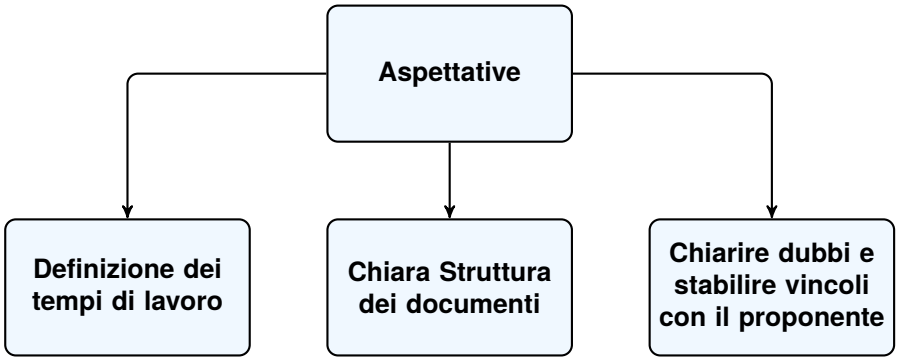
- Documento in dettaglio sull'organizzazione del lavoro
- Riconoscere se il materiale prodotto rispetta vincoli ed è di qualità
- Identificare competenze e strumenti necessari

2.1.2 Descrizione

Il processo primario di fornitura è quello nel quale vengono scelte le procedure e le risorse necessarie allo sviluppo del prodotto. La sezione seguente raccoglie tutte le norme che i membri del team CodeSix si impegna a seguire al fine svolgere in maniera soddisfacente l'attività di fornitura per la proponente* *Zucchetti S.p.A.* e i committenti* *Prof. Tullio Vardanega* e *Prof. Riccardo Cardin*.

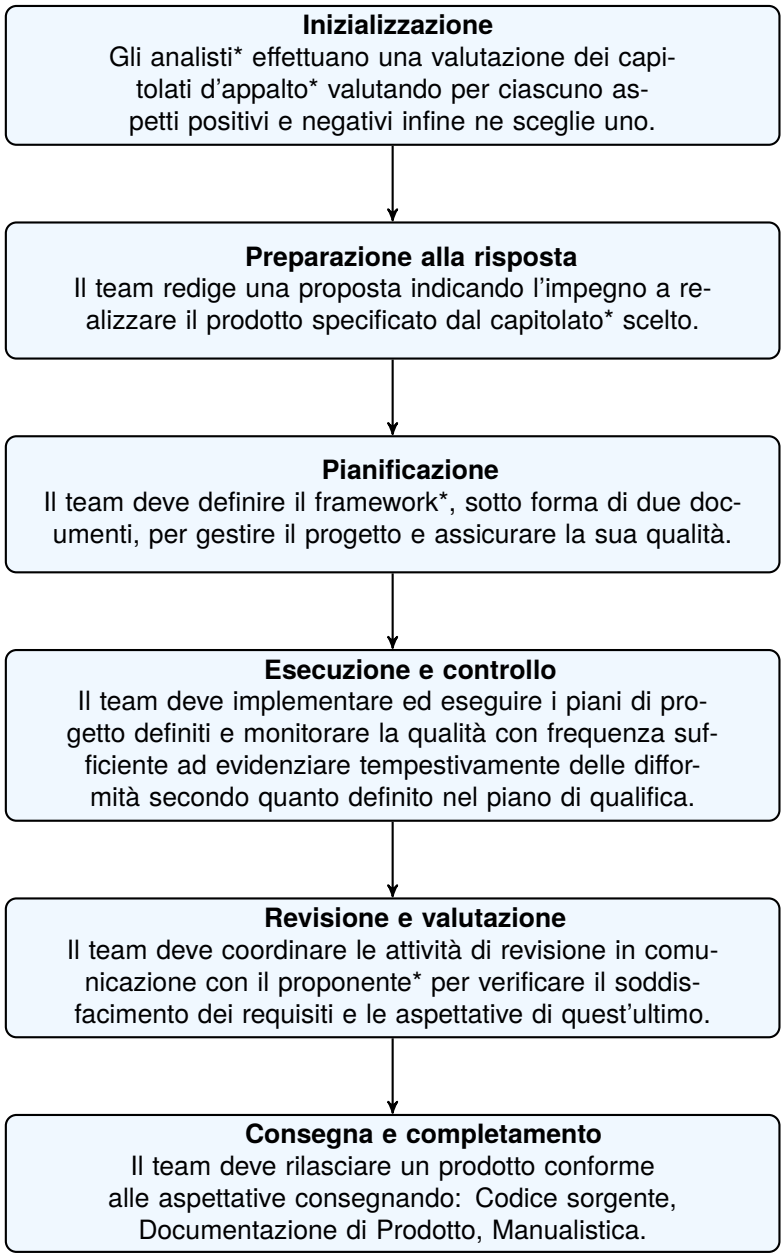
2.1.3 Aspettative

Le aspettative per l'attività di Fornitura possono essere così sintetizzate in un diagramma:



2.1.4 Composizione del processo

Per il processo di Fornitura vengono individuate le attività mostrate nel seguente diagramma



2.1.5 Rapporto con il proponente* - *Zucchetti S.p.A.*

Il rapporto che il gruppo mantiene con *Zucchetti S.p.A.* ha la funzione di raggiungere un accordo sui seguenti punti:

2.1.6 Materiale consegnato

Vengono qui elencati i documenti consegnati ai committenti *Prof. Tullio Vardanega* e *Prof. Riccardo Cardin*, e alla proponente *Zucchetti S.p.A.* Quest'ultimi assicurano la trasparenza delle attività di Analisi, Pianificazione, Verifica, Validazione* e Controllo di qualità per l'intero perdurare del ciclo di vita del progetto.

- ***Analisi dei Requisiti:*** documento che raccoglie l'analisi effettuata sui casi d'uso e dei requisiti, lo scopo del documento è di:
  - Definire nello specifico le funzionalità offerte dal prodotto;
  - Prevenire eventuali ambiguità sul capitolato.
- ***Piano di Progetto:*** documento che contiene la pianificazione preventiva dei tempi e delle attività, oltre all'analisi dei rischi, il consuntivo di periodo, la data e i costi previsti per la realizzazione del progetto finale;
- ***Piano di Qualifica:*** documento che raccoglie le modalità che vengono adottate in fase di verifica e validazione;
- ***Proof of Concept* e Technology Baseline*:*** costituiscono una vista ad alto livello, nel contesto della Progettazione, del prodotto che CodeSix andrà a realizzare e quali tecnologie utilizzerà per farlo;
- ***Glossario:*** raccoglie parole e terminologia utilizzata con un significato specifico o tecnico al fine di minimizzare la confusione dovuta alla loro presenza nel testo.

2.1.7 Preparazione al collaudo

Al fine di ottenere un esito positivo in fase di collaudo, il prodotto finito sarà sottoposto ad una batteria di test così articolata:

- Test d'unità;

- Test d'integrazione;
- Test di sistema;
- Test di accettazione.

Solamente al superamento con successo dei test sarà possibile affermare che il prodotto è infatti affidabile, corretto e completo. In sede di collaudo è necessario che:

- i test elencati nel *Piano di Qualifica 1.0.0* sono stati correttamente eseguiti riportando un esito conforme alle metriche specificate;
- le promesse fatte al proponente*, definite come requisiti obbligatori nell'*Analisi dei Requisiti 1.0.0*, sono state rispettate;
- nel caso siano stati implementati requisiti desiderabili e facoltativi, questi soddisfano le aspettative minime del proponente*.

2.1.8 Collaudo e consegna

Al momento della consegna, il gruppo CodeSix, si impegna a consegnare alla proponente* *Zucchetti S.p.A.* e ai committenti* *Prof. Tullio Vardanega* e *Prof. Riccardo Cardin* quanto segue:

- **Codice sorgente**;
- **Documentazione di prodotto**, secondo quanto già menzionato in §2.1.6, alla quale si deve aggiungere:
  - **Manuale Utente:** guida destinata all'utente finale per l'installazione e l'utilizzo del prodotto;
  - **Manuale dello Sviluppatore:** guida destinata a sviluppatori* esterni per consentire di espandere il progetto;
  - ***Glossario 1.0.0:*** documento che raccoglie la terminologia tecnica e/o quella a cui è associato un significato particolare per facilitare la comprensione dei documenti.

2.1.9 Metriche

Per il processo di fornitura non sono state individuate metriche qualitative specifiche.

2.1.10 Strumenti

Per il processo di fornitura sono stati utilizzati i seguenti strumenti:

- **Jira Software*** per gestire le task* da svolgere e i membri del gruppo a cui erano assegnate.

2.2 Sviluppo

2.2.1 Scopo

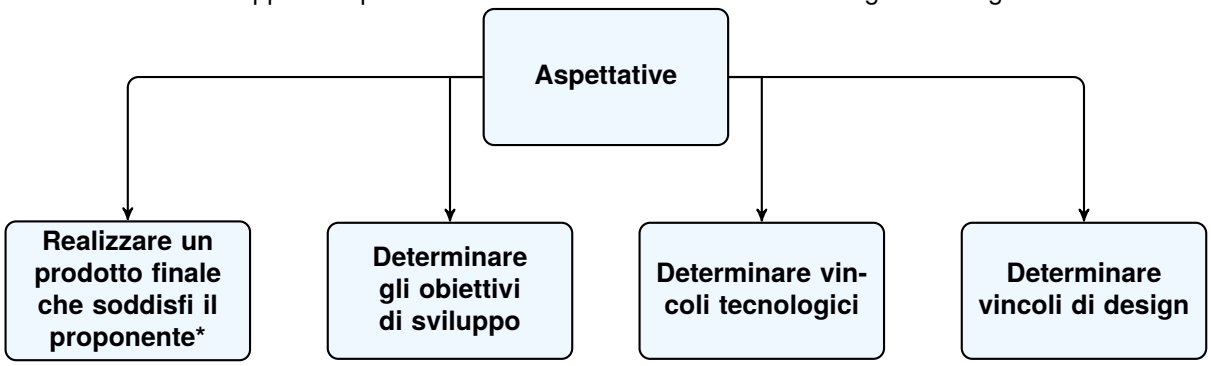
Il processo di sviluppo definisce le attività e i compiti necessari per ottenere il prodotto finale che soddisfi il proponente*.

2.2.2 Descrizione

Il processo di sviluppo include tutte le attività e i compiti dello sviluppatore* tra cui l'Analisi dei Requisiti, Progettazione, Codifica, Integrazione, Test, Installazione.

2.2.3 Aspettative

Per il Processo di Sviluppo le aspettative sono riassunte visivamente nel seguente diagramma:



2.2.4 Composizione del processo

Le sottosezioni seguenti normeranno ognuna delle attività caratterizzanti della fase di Sviluppo.

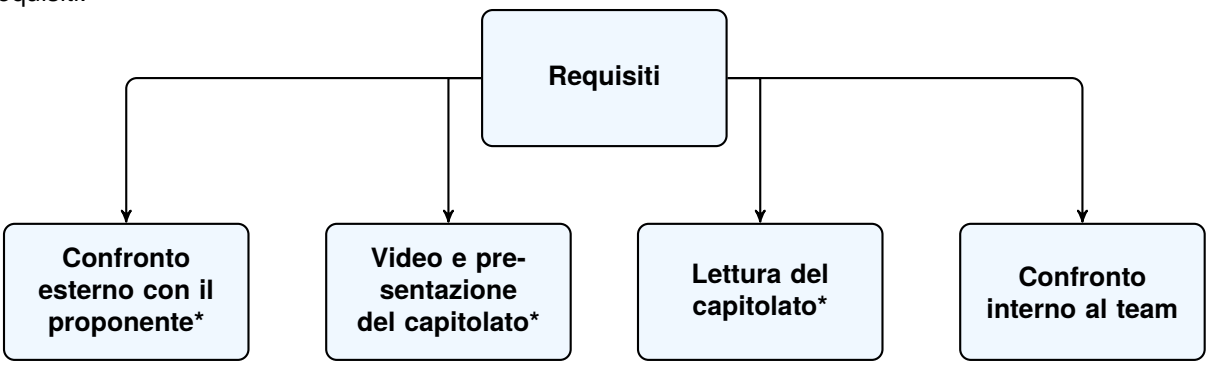
2.2.4.1 Analisi dei requisiti

2.2.4.1.1 Scopo

Durante l'attività di analisi dei requisiti viene redatta la documentazione formale contenente tutti i requisiti richiesti dal proponente*. Questo compito ricade sui membri del team che occupano il ruolo di Analista* i quali sono incaricati di individuare i requisiti diretti, indiretti, impliciti ed espliciti che sono richiesti dal proponente* per una realizzazione del prodotto soddisfacente.

2.2.4.1.2 Descrizione

Il diagramma sottostante riassume quali sono le fonti alle quali il team CodeSix attingerà per identificare i requisiti.



2.2.4.1.3 Aspettative

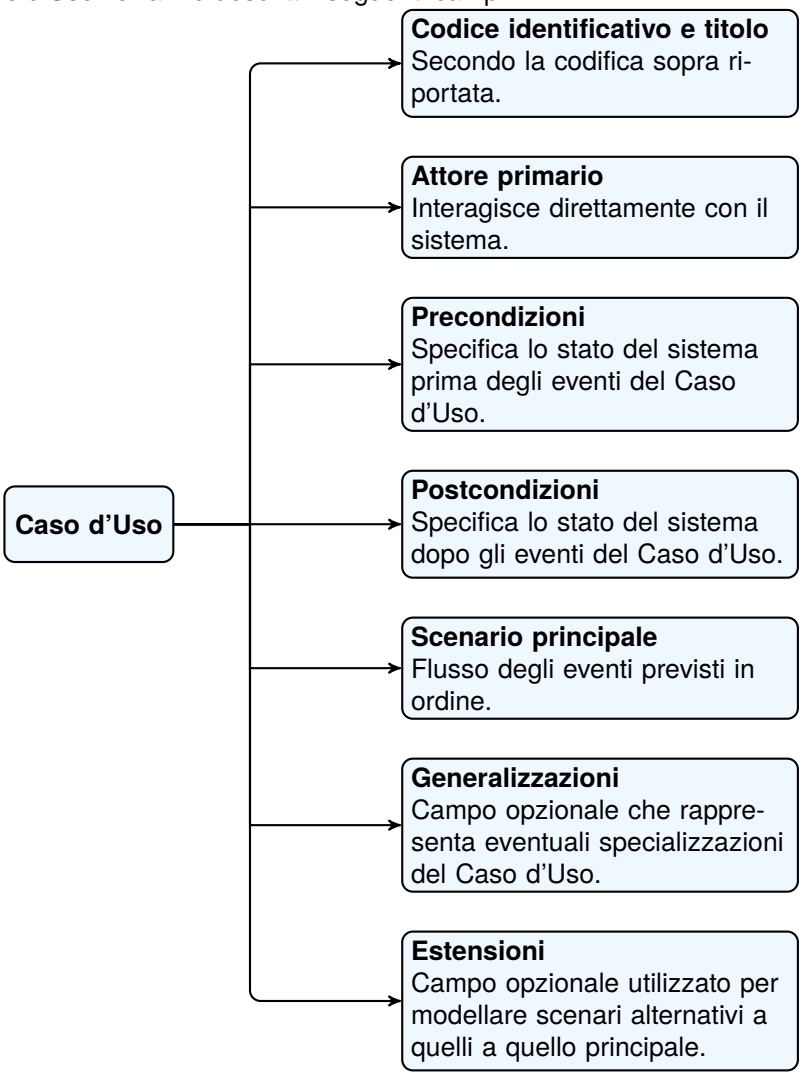
L'obiettivo dell'attività di analisi dei requisiti consiste nella creazione della documentazione formale contenente tutti i requisiti richiesti dal proponente*.

2.2.4.1.4 Casi d'uso

Si decide di seguire la seguente convenzione per la nomenclatura dei casi d'uso:

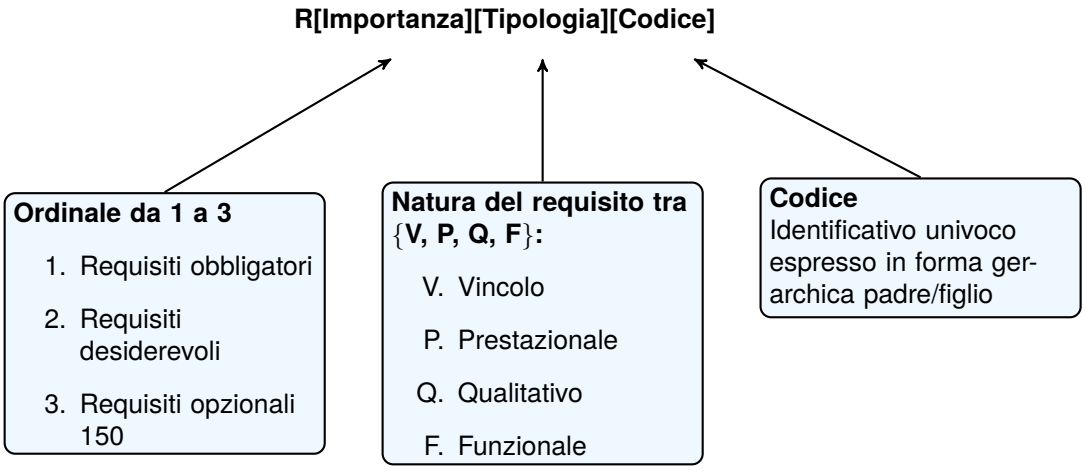
UC[Numero caso d'uso].([Numero caso d'uso figlio]) - [Titolo]

La struttura dei Casi d'Uso prevede che ci possa essere più di un livello di gerarchia figliare. Importante precisazione è che il titolo del caso d'uso deve essere relativo ad una funzionalità e non ad un evento. Per ogni Caso d'Uso verranno descritti i seguenti campi:



2.2.4.1.5 Requisiti

Per la rappresentazione dei requisiti il team CodeSix adotta la rappresentazione mostrata nel diagramma seguente:



La parte relativa al **Codice** può poi essere ulteriormente specificata nel seguente modo:

[CodiceBase].[CodiceSottoCaso]

Dove il **CodiceSottoCaso** è campo ulteriore opzionale sotto forma di ordinale a partire da 1 che identifica il sotto caso del requisito. La struttura dei requisiti può prevedere più livelli di gerarchia.

Le tabelle che riporteranno i Requisiti saranno organizzate nel seguente modo:

Codice	Classe	Descrizione	Fonti
Nella forma espressa sopra	Ripetuto per leggibilità	Descrizione sintetica del requisito	Origine del requisito

2.2.4.1.6 Tracciamento dei Requisiti

Al fine di mantenere il tracciamento tra i Requisiti e le fonti di quest'ultimi vengono aggiunte le seguenti tabelle.

Requisito	Fonte
R1F1	Fonte del Requisito
R1F2	Fonte del Requisito

Fonte	Requisiti individuati
UC1	R1F1, R1F2
Capitolato*	R1F3, R2F1

2.2.4.2 Progettazione

2.2.4.2.1 Scopo

Durante la fase di Progettazione vengono individuate le caratteristiche che il prodotto deve possedere per costituire la soluzione che meglio soddisfa le aspettative e i requisiti degli stakeholder.

2.2.4.2.2 Descrizione

L'attività di Progettazione deve occuparsi di:

- Garantire la qualità del prodotto sviluppato perseguendo la correttezza per costruzione e non per correzione;
- Organizzare e suddividere i compiti implementativi al fine di facilitare la codifica da parte dei Programmatori*;
- Ottimizzare tempo e risorse assegnate.

2.2.4.2.3 Aspettative

L'attività mira alla definizione della architettura di sistema, inizialmente avviandola con il ***Proof of Concept**** della ***Technology Baseline*** per poi approfondirla e meglio definirla nella ***Product Baseline****.

2.2.4.2.4 Technology Baseline*

Motiva le tecnologie, i framework* e le librerie selezionate per la realizzazione del prodotto. Sarà il Progettista ad occuparsene e dovrà contenere:

- Tecnologie adottate e le motivazioni dietro le scelte;
- Le relazioni tra ogni componente e il requisito che soddisfa, per avere un tracciamento;
- ***Proof of Concept***, ovvero un prototipo per dimostrare le funzionalità del prodotto.

2.2.4.2.5 Product Baseline*

Include i seguenti item:

- Design pattern* utilizzati, accompagnati da una descrizione;
- Diagrammi UML* delle classi, di attività, di sequenza e dei package;
- Una definizione delle classi, evitando nomi e funzionalità ridondanti;
- Tracciamento delle classi, in modo che ciascun requisito sia soddisfatto da una classe;
- Test di unità su ogni componente, in modo da verificare il corretto funzionamento.

2.2.4.3 Codifica

2.2.4.3.1 Scopo

Nell'attività di Codifica, svolta dal ruolo del Programmatore*, viene concretizzata la progettazione tramite l'effettiva realizzazione del software. Si occupa di trasformare in codice l'architettura definita dai Progettisti* a più alto livello.

2.2.4.3.2 Descrizione

La Codifica come attività deve rispettare gli obiettivi descritti dal ***Piano di Qualifica 1.0.0*** al fine di garantire che il codice prodotto sia di qualità. Nelle sezioni seguenti verranno inizialmente normate le convenzioni di carattere più generali in maniera agnostica al linguaggio di programmazione per poi proseguire con quelle specifiche dei linguaggi utilizzati, in particolare JavaScript*.

2.2.4.3.3 Aspettative

Il risultato di questa attività è la produzione del prodotto software che possegga le caratteristiche e i Requisiti stipulati con il proponente*. Il codice prodotto da questa attività dovrà anche rispettare le convenzioni per assicurarne la leggibilità a fini di manutenzione, espansione, verifica e validazione.

2.2.4.3.4 Convenzioni sulla lingua

Il team CodeSix decide di utilizzare la lingua inglese per nominare variabili, metodi, classi e commenti. Le motivazioni dietro la scelta sono le seguenti:

- Gli statement e le keywords dei linguaggi di programmazione sono in inglese, risulta quindi spontaneo e naturale utilizzare l'inglese per i commenti o per i nomi di variabili e metodi;
- Dal fatto che il progetto preveda la creazione di un ***Manuale dello Sviluppatore*** per chi volesse espandere il progetto si trae come naturale conseguenza che il codice debba risultare leggibile al pubblico più ampio possibile;
- Gli IDE* sono spesso forniti di strumenti atti alla rilevazione di errori ortografici con molto più supporto per la lingua inglese.

2.2.4.3.5 Convenzioni per la documentazione

- **Intestazione**: Ogni file sorgente consegnato sarà corredato di un intestazione sotto forma di commento descritta nello snippet* seguente:

```
/*
    Prova di intestazione del codice
*/
```

2.2.5 Metriche

2.2.5.1 Metriche per la manutenibilità

- **Sloc**: linee totali di codice;
- **Slocn**: linee totali di codice commenti esclusi;
- **Locom**: linee totali di commenti;

Codice	Nome	Descrizione	Formula
MPD2	Errori rilevati	Misura la percentuale di errori rilevati nel codice.	(Sloc errate / Sloc) *100
MPD4	Comprensibilità del codice	Misura la percentuale di linee di commenti.	(Locom / Sloc) * 100

2.2.5.2 Metriche per la portabilità

- $\mathbf{B_{sup}}$: browser supportati;
- $\mathbf{B_{ric}}$: browser richiesti.

Codice	Nome	Descrizione	Formula
MPD8	Ambienti di esecuzione supportati	La percentuale di browser supportati dal prodotto.	(B_{sup} / B_{ric})*100

2.2.5.3 Metriche per la funzionalità

- $\mathbf{N_{rm}}$: numero di requisiti non implementati;
- $\mathbf{N_{ri}}$: numero di requisiti totali individuati.

Codice	Nome	Descrizione	Formula
MPD1	Aderenza requisiti funzionali	Descrive la capacità percentuale di aderenza delle funzioni sviluppate rispetto a quelle richieste.	(N_{rm} / N_{ri})*100

2.2.5.4 Metriche per l'affidabilità

Codice	Nome	Descrizione	Formula
MPD6	Errori inattesi	Capacità di evitare errori o risultati inattesi in fase di esecuzione.	(Numero errori rilevati)

2.2.5.5 Metriche per l'usabilità

Codice	Nome	Descrizione	Formula
MPD7	Facilità di utilizzo	Capacità del prodotto di permettere all'utilizzatore di raggiungere obiettivi specifici in modo corretto e completo in uno specifico contesto.	(Numero medio di click per funzionalità)

2.2.6 Strumenti

- **D3.js***: Libreria per la manipolazione del Document Object Model (DOM) di una pagina web. Mette a disposizione molti tipi di grafici richiesti dal progetto;
- **DruidJs***: Libreria che fornisce algoritmi per la riduzione dimensionale;
- **Node.js***: runtime JavaScript largamente utilizzata per la creazione di applicazione di rete;

- **Npm***: package manager per il linguaggio JavaScript;
- **React.js***: Libreria JavaScript per la creazione di UI, miglioramento delle performance di renderizzazione delle pagine, facilitazione di manutenibilità e testing;
- **Visual Studio Code***: IDE performante ed estendibile scelto dal gruppo per lo sviluppo del codice.
- **Papa Parse***: Libreria potente e robusta che permette di gestire efficientemente il parsing di file CSV di grandi dimensioni su pagine web sia in locale che sulla rete.

3 Processi di Supporto

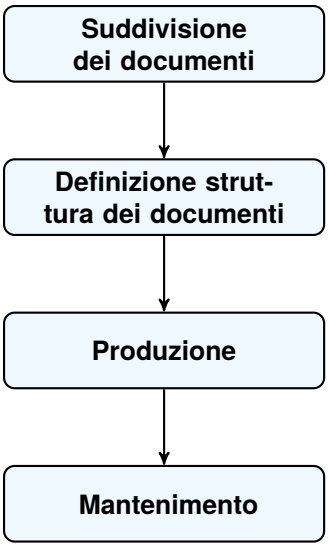
3.1 Documentazione

3.1.1 Scopo

Ogni processo e attività significativi per lo sviluppo del progetto saranno documentati. Nella presente sezione verranno descritte le regole e gli standard che disciplineranno il processo di Documentazione durante l'intero ciclo di vita del software, permettendo di ottenere prodotti documentali coerenti e validi dal punto di vista tipografico e formale.

3.1.2 Descrizione

Ad ogni stesura si vogliono applicare le medesime norme interne e convenzioni per la stesura, la verifica e l'approvazione della documentazione sia interna che esterna. La normativa della Documentazione è quindi così composta:



3.1.3 Aspettative

Le aspettative per questo processo sono le seguenti:

- Individuare una struttura comune per tutti i prodotti del processo, nell'arco del ciclo di vita del software;
- Raccogliere tutte le norme comuni per la stesura dei documenti ufficiali.

3.1.4 Ciclo di vita di un documento

Ogni documento prodotto dal gruppo CodeSix segue le seguenti fasi di ciclo di vita.

- **Template**: viene inizializzato il template del documento come descritto a §4.1.7.2 "Struttura dei documenti - template";
- **Strutturazione**: viene stesa una scaletta preliminare elencante i macro argomenti da trattare e sviluppare;
- **Stesura**: i membri del gruppo a cui è stato assegnato il compito di redazione dei documenti suddividono le singole sezioni da svolgere e procedono secondo una stesura incrementale;

- **Verifica**: ogni sezione redatta viene quindi sottoposta al processo di verifica da uno o più membri responsabili di tale compito;
- **Approvazione**: al termine della verifica il Responsabile di Progetto* ne stabilisce la validità ponendo il documento come completo e rendendolo disponibile alla consultazione.

3.1.5 Classificazione documentale

I documenti prodotti saranno categorizzati nel seguente modo:

- **Documenti interni**:
  - **Verbali interni**: contengono il resoconto delle discussioni, argomenti affrontati e decisioni prese internamente al gruppo;
  - **Norme di Progetto**: contiene norme e regole a cui tutti i membri del gruppo si devono attenere per tutto il ciclo di vita del progetto.
- **Documenti esterni**:
  - **Piano di Qualifica**: definisce le caratteristiche di qualità che il prodotto deve rispettare e i criteri di valutazione a cui il gruppo si attiene per la verifica;
  - **Piano di Progetto**: espone la pianificazione del progetto in tutti i suoi processi, il preventivo sui costi previsti e l'impegno orario dei membri del gruppo;
  - **Analisi dei Requisiti**: espone i requisiti che il prodotto finale deve rispettare;
  - **Glossario**: contiene le definizioni di termini rilevanti con significati particolari o utilizzati in maniera specifica;
  - **Verbali esterni**: contiene il resoconto delle discussioni e degli argomenti affrontati dal gruppo con il proponente* e/o il committente*.

3.1.6 Nomenclatura documentale

Ogni documento prodotto viene denominato utilizzando la convenzione CamelCase senza spazi intermedi e secondo i nomi ufficiali nel seguente modo:

[NomeDocumento]_v[X].[Y].[Z]

dove 'v' indica il numero di versione. Allo stesso modo ogni verbale verrà nominato come segue

[VI/VE]_[YYYY]_[MM]_[DD]

In particolare:

- Per la definizione della parte v[X].[Y].[Z] del nome del documento si veda §4.2.4 "Gestione della configurazione - Versionamento";
- Per la definizione della parte [YYYY].[MM].[DD] del nome del verbale si veda § 4.1.9 "Documentazione - Norme Tipografiche".

3.1.7 Struttura dei documenti

3.1.7.1 Template

Al fine di standardizzare ogni documento e per permettere ai redattori di occuparsi specialmente della documentazione, è stato creato un template in LaTeX che genera la prima pagina, il registro delle modifiche e l'indice, descritti come di seguito.

3.1.7.2 Documentazione

Ogni documento redatto sarà composto dai seguenti elementi strutturali:

3.1.7.2.1 Prima pagina

- Logo del gruppo CodeSix;
- [Nome del documento] - Progetto Ingegneria Del Software;
- Versione del documento
- Elenco membri del gruppo;
- Dipartimento di Matematica Università degli Studi di Padova
- Data ultima validazione.

3.1.7.2.2 Registro delle modifiche

Riassume in tabella le modifiche apportate al documento nel seguente modo:

Versione	Modifica	Ruolo	Esecutore	Data

3.1.7.2.3 Indice

Elenco puntato delle sezioni e sottosezioni trattate nel corpo del documento con riferimenti al testo.

3.1.7.2.4 Pagina di testo

- Header: composto da
  - Nome del gruppo;
  - Nome del documento.
- Corpo del documento
- Footer: composto da
  - Numero di pagina / Totale pagine.

3.1.7.3 Verbali

Ogni verbale interno ed esterno, oltre alla prima pagina e al registro delle modifiche come esposti in precedenza, dovranno contenere le seguenti informazioni al fine di essere conciso e schematico per una facile consultazione:

- Luogo dell'incontro;
- Durata dell'incontro;
- Partecipanti;
- Obiettivi dell'incontro: elenco degli obiettivi per l'incontro;
- Argomenti trattati: contiene gli argomenti discussi in concomitanza degli obiettivi del punto precedente;
- Decisioni: elenco delle scelte prese durante l'incontro.

3.1.8 Norme tipografiche

Sono di seguito esposte le norme tipografiche e gli stili adottati per garantire la congruenza fra i documenti prodotti.

- **Elenchi**
  Gli elenchi puntati e numerati seguono le seguenti norme:
  - Ogni voce comincia per lettera maiuscola ed è contrassegnato da un elemento che la distingue;
  - Ogni voce termina con l'inizio della successiva;
  - L'ultima voce termina con l'inizio di una nuova sezione;
  - Ogni sottovoce annidata viene contrassegnata da un elemento differente rispetto a quella del padre.
- **Data**
  Per la rappresentazione della data il gruppo ha scelto il seguente formato:

  [YYYY].[MM].[DD]

  dove:
  - [YYYY] indica l'anno a quattro cifre;
  - [MM] indica il mese;
  - [DD] indica il giorno;
- **Riferimenti di glossario** Al fine di minimizzare le ambiguità è stato messo a disposizione il Glossario, il quale raccoglie le definizioni di tutti i termini importanti o con un significato specifico presenti nella documentazione. Questi sono individuati nel testo da '*' ad apice e rimandano alla versione di glossario corrispondente a quella del documento in questione.
- **Elementi grafici**
  - Tabella;
  - Figura;

  gni elemento deve essere corredato dalla stringa formata nel modo che segue:

  Elemento[X]: [Descrizione]

  dove:
  - Elemento: il corretto elemento grafico dalla lista sopracitata;
  - [X]: numero progressivo a partire da 1 per ogni tipologia di elemento.
  - Descrizione: descrizione dell'elemento;
- **Sigle**
  Per i ruoli che i membri del gruppo svolgono a rotazione all'interno del progetto si adottano le seguenti:
  - RE : Responsabile di Progetto;
  - AM : Amministratore;
  - AN : Analista;
  - PT : Progettista;
  - PR : Programmatore;

- VE : Verificatore.

Per il documento di Analisi dei Requisiti in riferimento all'importanza sono usate:

- OB : Obbligatorio;
- DE : Desiderabile;
- OP : Opzionale.

3.1.9 Metriche

Per il processo di Documentazione sono state individuate le seguenti metriche:

- $\mathbf{N_{frasi}}$: numero di frasi;
- $\mathbf{N_{lettere}}$: numero di lettere;
- $\mathbf{N_{parole}}$: numero di parole;

Codice	Nome	Descrizione	Formula
MPD5	Apprendibilità	L'indice di Gulpease permette di capire il grado di leggibilità di un documento in lingua italiana.	$\frac{80+[(300*N_{frasi})-10*(N_{lettere})]}{N_{parole}}$
MPD3	Errori documentali	Misura il numero di errori ortografici o di aderenza alle norme rilevati.	-

3.1.10 Strumenti

Per la produzione dei documenti richiesti, il gruppo CodeSix ha deciso di usare come supporto tipografico gli strumenti:

- **Overleaf***: editor collaborativo basato su cloud utilizzato per scrivere, modificare e pubblicare documenti scientifici che sfrutta LaTeX, linguaggio di markup per la preparazione di testi al fine di automatizzare la gran parte della composizione tipografica di un documento;
- **Draw.io***: software di disegno grafico multipiattaforma gratuito e open source sviluppato in HTML5 e JavaScript. La sua interfaccia può essere utilizzata per creare diagrammi come diagrammi di flusso, wireframe, diagrammi UML, organigrammi e diagrammi di rete.

3.2 Gestione della configurazione

3.2.1 Scopo

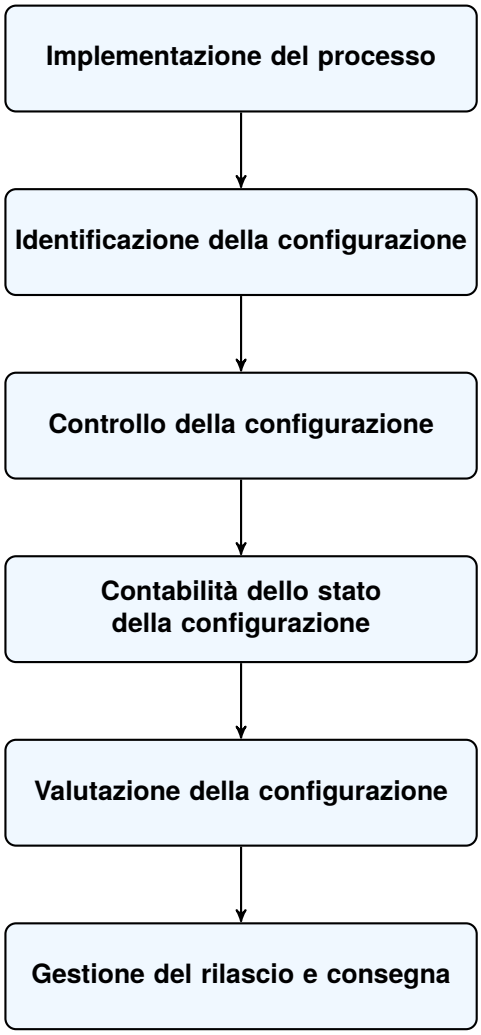
La gestione della configurazione è un processo che pone come obiettivo la gestione della produzione del codice e dei documenti in maniera ordinata e metodica al fine di renderli accessibili alle parti interessate, attribuendo un sistema di versionamento appropriato e una posizione specifica nel repository*.

3.2.2 Aspettative

- Identificare, definire e distribuire i pacchetti software in un sistema;
- Gestire le modifiche e le release dei pacchetti;
- Controllare lo stato delle modifiche e le richieste di modifica;
- Assicurare consistenza, sicurezza e correttezza dei pacchetti;
- Controllare l'archiviazione, la gestione e la consegna dei pacchetti.

3.2.3 Descrizione

Il processo si compone delle seguenti attività:



3.2.4 Versionamento

3.2.4.1 Codice di versionamento

Ogni documento nel suo ciclo di vita attraversa varie versioni di produzione che vengono identificate da un valore:

$$[X].[Y].[Z]$$

dove:

- [X]: indica la versione finale approvata dal Responsabile di Progetto e resa disponibile alla consultazione. Il valore comincia da 0;
- [Y]: indica una revisione completa del documento attraverso il processo di verifica. Il valore comincia da 0 e si azzera quando [X] si aggiorna;
- [Z]: indica una modifica incrementale minore con verifica relativa. Il valore è crescente e comincia da 0 ogni volta che viene aggiornato X o Y.

3.2.5 Tecnologie adottate

La gestione delle versioni viene effettuata tramite un repository* remoto, con sistema di versionamento Git* e conservato su GitHub*.

3.2.6 Modifiche alle repository*

Ogni membro del gruppo può modificare quando vuole i file contenuti nella repository* ad eccezione di quelli contenuti nel branch* Main, per il quale dovrà essere creata una **Pull Request*** che dovrà essere approvata da uno o più membri del gruppo, diversi dal modificatore. Le modifiche di lieve impatto possono essere eseguite autonomamente, mentre le modifiche importanti richiedono l'approvazione del Responsabile di Progetto e la loro giustificazione all'interno team.

3.2.7 Metriche

Il processo di Gestione della Configurazione non prevede l'utilizzo di metriche particolari.

3.2.8 Strumenti

Per il processo di Gestione della Configurazione non sono stati individuati strumenti specifici.

3.3 Gestione della Qualità

3.3.1 Scopo

Il gruppo CodeSix ha deciso di sviluppare il prodotto software secondo i tre assi di intervento del sistema qualità al fine di:

- Fissare politiche e obiettivi di qualità in relazione a processi e risorse necessari;
- Verificare al fine di assicurare che il prodotto sia conforme alle attese;
- Attuare un miglioramento continuo sulla base di queste ultime.

3.3.2 Piano di Qualifica

Nel documento ***Piano di Qualifica*** il gruppo CodeSix illustra come intende perseguire le qualità di prodotto e di processo stabilendo metriche opportune ed adeguate sulla base di standard e definendo i test da eseguire per il processo di verifica al fine di attuare un miglioramento continuo. Per garantire la qualità dei processi abbiamo fatto riferimento allo standard **ISO/IEC/IEEE 12207:1995**; per la qualità di prodotto è stato fatto riferimento allo standard **ISO/IEC 9126**.

3.3.3 Struttura descrittiva

Il seguente paragrafo illustra lo schema generale di descrizione delle metriche a seguito degli attributi di qualità scelti utilizzando tabelle che sintetizzano quanto esposto in tale modo: Per la sezione relativa alla qualità di processo

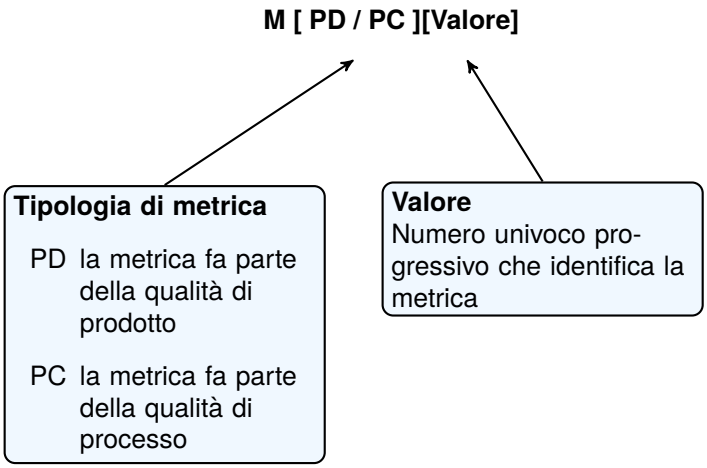
Processo	Descrizione	Metriche
	Processi primari	
	Processi di supporto	
	Processi organizzativi	

Per la sezione relativa alla qualità di prodotto:

Obiettivo	Descrizione	Metriche
	Metriche interne	
	Metriche esterne	

3.3.4 Codici identificativi delle metriche

Tutte le metriche che sono definite nel Piano di Qualifica sono identificate univocamente dalla seguente convenzione:



3.3.5 Struttura descrittiva delle metriche

Alle sezioni di qualità di prodotto e di processo sono aggiunte delle strutture tabellari che descrivono le metriche scelte:

Codice	Nome Metrica	Descrizione

Codice	Valore accettabile	Valore ottimale

3.3.6 Metriche

Il processo di Gestione della Qualità non prevede l'utilizzo di metriche particolari.

3.3.7 Strumenti

Per il processo di Gestione della Qualità non sono stati individuati strumenti specifici.

3.4 Verifica

3.4.1 Scopo

Lo scopo del processo di verifica è quello di assicurare che tutte le attività, realizzate in un dato segmento di ciclo di vita del prodotto software, non abbiano introdotto errori e che rispettino tutti i requisiti necessari. Questo processo viene applicato ad ogni fase del progetto e porta quindi all'attivazione di due attività di analisi: quella statica e quella dinamica.

3.4.2 Attività

Attività	Descrizione	Metodi di Lettura
Analisi Statica	Studia la documentazione e il codice non eseguibile occupandosi di accertare la conformità alle regole e l'assenza di errori basandosi sulle indicazioni presenti nel documento Piano di Qualifica.	• Inspection • Walkthrough
		Test
Analisi dinamica	Opera sul prodotto software in esecuzione tramite un insieme di test verificandone la correttezza e l'assenza di errori e anomalie. I test eseguiti devono poter essere ripetibili e automatizzati.	• Test di Unità • Test di Integrazione • Test di Sistema • Test di Accettazione

Table 11: Caption

3.4.3 Metodi di lettura

Metodo	Descrizione	Agenti	Modalità
Inspection	Verifica la presenza di errori o anomalie procedendo in maniera mirata sulla documentazione e/o codice.	Verificatori	• Pianificazione • Lista di controllo (elenco selettivo) • Lettura • Correzione di errori
Walkthorugh	Verifica il prodotto procedendo secondo una lettura critica ad ampio spettro.	Verificatori Sviluppatori Redattori	• Pianificazione • Lettura • Discussione • Correzione

Table 12: Caption

3.4.4 Test

I test sono la componente necessaria per svolgere l'attività di analisi dinamica. Ogni test deve poter essere:

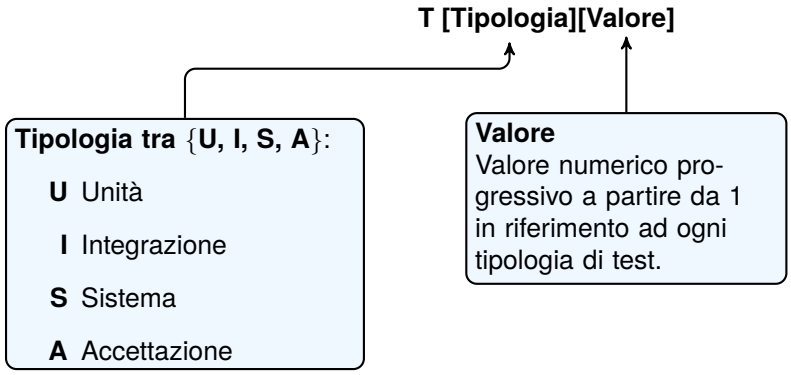
- **Ripetibile**: il test deve dare sempre lo stesso output a parità di insieme di dati di input. Per questo esso definisce un ambiente di esecuzione, il relativo stato iniziale e le procedure coinvolte;
- **Automatizzato**: i test eseguiti devono poter essere eseguiti in maniera automatizzata al fine di limitare il controllo umano e sfruttare in maniera efficiente le risorse disponibili per il processo di verifica.

3.4.5 Tipologia di Test

Test	Descrizione	Esempi di errori rilevabili
Unità	Testa le singole unità di codice ossia quelle parti che sono composte da uno o più moduli, ossia le componenti elementari dell'architettura di dettaglio	• limiti di iterazioni • flusso di esecuzione • valori di variabili
Integrazione	Agisce per verifica incrementale del sistema operando sulle singole componenti sviluppate in parallelo	• errori nelle componenti • modifica di interfacce • modifica ai requisiti • riuso di componenti inadatte • integrazione con altre applicazioni poco conosciute
Sistema	Per accertare il comportamento dinamico del prodotto finale rispetto ai requisiti sul lato fornitore.	• requisiti mal formulati • aspettative non rispettate
Accettazione	Accerta il soddisfacimento dei requisiti alla presenza del committente	

3.4.6 Tracciamento dei test

Ogni test viene identificato secondo un codice formato nel seguente modo:



3.4.7 Metriche

Qui riportati alcuni parametri necessari per la comprensione delle formule di seguito:

- **Budget at Completion (BAC)**: corrisponde al budget totale allocato per il progetto al suo inizio;
- **Requisiti Cambiati (RC)**: numero di requisiti che hanno subito una o più modifiche;
- **Requisiti Eliminati (RE)**: numero di requisiti che sono stati eliminati;
- **Requisiti Aggiunti (RA)**: numero di requisiti che sono stati aggiunti;
- **Requisiti Iniziali (RI)**: numero di requisiti iniziali;

- **Requisiti Soddisfatti (RS)**: numero di requisiti soddisfatti;
- **Requisiti Totali (RT)**: numero totale di requisiti attuali;
- **Linee di Codice Eseguite (LCE)**: numero di linee di codice che vengono eseguite dai test automatici;
- **Linee di Codice Totali (LC)**: numero di linee di codice totali;
- **Metriche di Qualità Soddisfatte (MQS)**: numero di metriche di qualità soddisfatte;
- **Metriche di Qualità Totali (MQT)**: numero di metriche di qualità totali;
- **Test Passati (TP)**: numero di test automatici passati;
- **Test Totali (TT)**: numero di test totali.

Codice	Nome Metrica	Descrizione	Formula
MPC1	Schedule Variance (SV)	Descrive se il gruppo sta aderendo ai tempi prestabiliti.	$(\frac{\text{EV}}{\text{PV}} - 1) * 100$
MPC2	Budget Variance (BV)	Differenza tra costo effettivo e budget preventivato.	$(\frac{\text{EV}}{\text{AC}} - 1) * 100$
MPC3	Actual Cost (AC)	Costo totale sostenuto per il lavoro effettivo completato fino ad ora.	-
MPC4	Earned Value (EV)	Quantificazione del valore del lavoro effettivamente svolto fino ad una certa data.	$\%$ completamento $*$ ETC
MCP5	Planned Value (PV)	Quantificazione del valore del lavoro che si dovrebbe aver raggiunto fino al momento del calcolo.	$\%$ lavoro pianificato $*$ BAC
MPC6	Estimate to Complete (ETC)	Costo previsto per completare tutto il lavoro del progetto rimanente.	-
MPC7	Estimate at Completion (EAC)	Attuale aspettativa del costo totale alla fine di un progetto.	AC + ETC
MPC8	Requirements stability index (RSI)	Confronta i requisiti attuali dopo eventuali aggiunte/eliminazioni/modifiche e i requisiti originali.	$(1 - \frac{\text{RC}+\text{RE}+\text{RA}}{\text{RI}}) * 100$
MPC9	Satisfied obligatory requirements (SOR)	Percentuale di requisiti obbligatori soddisfatti.	$\frac{RS}{RT} * 100$

MPC10	Code Coverage (CC)	Misura di quante righe/blocchi/archi del codice vengono eseguiti durante l'esecuzione dei test automatici.	$\frac{LCE}{LC} * 100$
MPC11	Passed test cases percentage (PTCP)	Misura la percentuale di test case passati dal codice.	$\frac{TP}{TT} * 100$
MPC12	Quality Metrics Satisfied (QMS)	Descrive la percentuale di metriche di qualità soddisfatte.	$\frac{MQS}{MQT} * 100$
MPC13	Non-calulated Risk	Indica il numero di rischi non preventivati.	-

3.4.8 Strumenti

Gli strumenti utilizzati per la verifica della documentazione riguardano attualmente l'indice di Gulpease. Con il fine di automatizzare e velocizzarne il relativo calcolo si è fatto uso dello strumento alla seguente **repository GitHub**

3.5 Validazione

3.5.1 Scopo

Il processo di validazione è un processo per determinare se i requisiti, il sistema finale e il prodotto software, soddisfano l'uso previsto specifico. Quando il gruppo ritiene di avere un prodotto pronto alla validazione effettua con il committente il collaudo del prodotto che consiste nella ripetizione dei test svolti con successo fino a quel momento.

3.5.2 Obiettivi

Il gruppo CodeSix intende quindi perseguire i seguenti obiettivi:

- Assicurare la correttezza dei test svolti durante il processo di verifica;
- Accertare la conformità del prodotto finale alle attese e alle specifiche dei requisiti.

3.5.3 Attività

La validazione del codice si compone di due attività:

- **Test di valutazione**: insieme delle attività svolte all'interno del gruppo;
- **Collaudo**: Attività di controllo che viene svolta durante la Revisione di Accettazione alla presenza del committente. L'obiettivo è dimostrare di aver implementato i requisiti richiesti, per farlo è fondamentale tracciare i test e i requisiti che validano, in modo da facilitare la verifica.

Il processo di validazione ha diverse attività:

- Progettazione e tracciamento dei test per il codice;
- Avvio dei test;
- Modifica al piano di qualifica;
- Controllo che il sistema soddisfi i requisiti.

3.5.4 Responsabilità dei test

Le figure che lavorano ai test e alla validazione sono:

- **Progettisti**: progetta e implementa i test;
- **Verificatori**: esegue i test e modifica il piano di qualifica aggiungendo i risultati ottenuti;
- **Responsabile di Progetto**: analizza i risultati dei test contenuti del Piano di Qualifica e decide se il risultato ottenuto è sufficiente per essere accettato, oppure ripetere le operazioni di testing.

3.5.5 Metriche

Il processo di Validazione non prevede l'utilizzo di metriche particolari.

3.5.6 Strumenti

Per il processo di Validazione non sono stati individuati strumenti specifici.

4 Processi Organizzativi

4.1 Organizzazione interna

4.1.1 Scopo

Lo scopo della seguente sezione è quello di definire l'organizzazione dei vari processi durante tutto il ciclo di vita. In particolare:

- La definizione di un modello di sviluppo adeguato e coerente per tutta la durata del processo;
- La suddivisione dei ruoli e la comunicazione interna ed esterna;
- Definire un quadro economico tenendo conto delle attività da svolgere, dei costi di ogni ruolo e di eventuali rischi che possono verificarsi in corso d'opera.

4.1.2 Aspettative

Le aspettative sono quindi quelle di pianificare al meglio i vari processi in modo da garantire efficacia ed efficienza lungo tutto il ciclo di vita del prodotto. Inoltre, in questa sezione si vuole definire al meglio quali sono le caratteristiche di ogni ruolo e gli strumenti per l'organizzazione e la comunicazione del team.

4.1.3 Ruoli

Per quanto riguarda la gestione dei ruoli, come previsto dal Regolamento di Progetto, ogni componente del team a rotazione ricoprirà almeno una volta ciascuna delle seguenti posizioni:

- ***Responsabile di Progetto***: è una figura molto importante per quanto riguarda la comunicazione tra il team e il proponente. Inoltre ha il compito di pianificare volta per volta il lavoro da svolgere. Esso infatti:
  - Definisce ed assegna i vari task* ai membri del team;
  - Assegna volta per volta i vari ruoli;
  - Gestisce il preventivo dei costi in base alle ore svolte, ai ruoli e ai possibili rischi.
- ***Amministratore di Progetto***: il compito dell'amministratore di progetto è quello di gestire e definire le varie componenti dell'ambiente di lavoro. Oltre alle infrastrutture per la gestione delle comunicazioni, sono fondamentali eventuali framework* e linguaggi per la fase di sviluppo e strumenti per la fase di test;
- ***Analista***: figura fondamentale nella parte dello studio del problema. Segue attivamente il processo di identificazione, definizione e tracciamento dei requisiti nella parte di Analisi;
- ***Progettista***: il ruolo del Progettista è strettamente legato a quello dell'Analista. Il suo compito è quello di definire il design della soluzione al problema a partire dai requisiti definiti dall'Analista. Per questo motivo sono richiesti uno studio e una conoscenza approfondita di pattern* architetturali e tecnologie in modo da favorire il riutilizzo di soluzioni già esistenti che si adattano al problema in esame;
- ***Programmatore***: la figura del Programmatore ha il compito, a partire dal design e dalle indicazioni del Progettista, di implementare le varie componenti del software rispettando i vincoli riportati nell'Analisi dei Requisiti. Inoltre deve definire i test (unità, integrazione, sistema e accettazione) in modo da semplificare e rendere automatica la fase di verifica;

- ***Verificatore***: la figura del Verificatore è cruciale per tutta la durata del progetto. Ogni modifica al codice e alla documentazione dovrà essere approvata del Verificatore prima di essere inclusa nella versione corrente di un particolare file. Per quanto riguarda la documentazione le modalità di verifica eseguita su di essa saranno:
  - *Walkthrough*: lettura completa del documento alla fine di evidenziare errori.
  - *Inspection*: lettura focalizzata sulle parti del documento che sono più frequentemente oggetto di errore o che richiedono maggiore attenzione nella stesura come per esempio le metriche e l'analisi dei requisiti.

4.1.4 Incontri e comunicazioni

Per quanto riguarda la gestione delle comunicazioni e dei meeting verranno utilizzati diversi strumenti. Per gli incontri interni:

- ***Telegram****: per le comunicazioni rapide tra i membri del team è stato deciso di utilizzare un gruppo Telegram*;
- ***Discord****: per gli incontri interni è stato aperto un canale su Discord* dove il team può riunirsi per discutere del progetto e collaborare. Inoltre sono stati aperti dei threads* dedicati a vari aspetti del progetto (Analisi dei Requisiti, Norme di Progetto, ...) dove è possibile caricare materiale e documentazione utile da consultare in un secondo momento.

Per gli incontri esterni:

- ***Email***: Per contattare il proponente per dei chiarimenti sul capitolato o per fissare un meeting verrà utilizzata l'email del team **codesix.swe@gmail.com**;
- ***Zoom****: Per gli incontri esterni con il proponente si è deciso di utilizzare Zoom* in modo da rendere più semplice la condivisione di materiale attraverso l'opzione di condivisione dello schermo e delle chat per l'invio di file.

Al termine di ogni meeting sia interno che esterno verrà redatto un verbale riepilogativo dell'incontro appena svolto. Inizialmente il documento verrà redatto utilizzando i template messi a disposizione da Jira* Software, e in un secondo momento verrà prodotto un documento Latex* finale. La struttura del documento è normata alla §4.1.7.3.

4.2 Gestione infrastrutture

4.2.1 Scopo

Lo scopo della seguente sezione è quello di andare a definire gli strumenti e le modalità per la pianificazione, la comunicazione e la gestione dell'ambiente di lavoro.

4.2.2 Aspettative

Ciò che ci si aspetta da questa sezione è di andare a definire, in maniera chiara e concorde tra tutti i membri del team, come verranno organizzati tutti gli strumenti di comunicazione e pianificazione. Ci si aspetta inoltre che tali prassi vengano messe in pratica da tutti i componenti del team in ogni fase di progetto.

4.2.3 Strumenti

4.2.3.1 Comunicazione

In questa sottosezione verranno descritte più approfonditamente le tecnologie di comunicazione già menzionate nella §5.1.4 "Organizzazione Interna - Incontri e comunicazioni".

4.2.3.1.1 Telegram*

Per quanto riguarda la comunicazione interna al gruppo si è deciso di utilizzare Telegram*, un'applicazione di messaggistica istantanea basata su cloud. Viene utilizzata giornalmente dal team per le comunicazioni rapide, per aggiornamenti sullo svolgimento dei lavori o per la collaborazione tra membri. Tra le caratteristiche che ci hanno portato a scegliere Telegram* rispetto ad altri competitor ci sono:

- Il salvataggio delle chat criptate sul cloud che garantisce una maggiore sicurezza e prestazioni migliori;
- La possibilità di usufruire di chat vocali con sistema push to talk e di videochiamate;
- La possibilità di utilizzare feature utili come sondaggi, bot e altre.

4.2.3.1.2 Discord*

Come già accennato nella sezione precedente per gli incontri interni tra i membri del team si è optato per Discord* che è una piattaforma di messaggistica VoIP e messaggistica istantanea. Alcune caratteristiche e feature che hanno indirizzato il team verso questa scelta sono:

- La bassa latenza che garantisce prestazioni migliori rispetto ad altri competitor;
- La possibilità di utilizzare feature utili come threads* testuali, bot e altre.

Per una migliore organizzazione del canale Discord* è stato deciso di creare diversi threads* dedicati a diversi ambiti di progetto (Analisi dei requisiti, Norme di progetto, ecc...) dove il team può discutere e lavorare su tale argomento. Questo garantisce, oltre ad un ambiente di lavoro più ordinato, anche una maggiore facilità nella ricerca di materiale utile condiviso nelle varie chat.

4.2.3.1.3 Gmail*

Per quanto riguarda invece le comunicazioni esterno col proponente è stato deciso di aprire un' account Google per poter usufruire del servizio di posto elettronica Gmail*. La mail verrà utilizzata essenzialmente per porre eventuali dubbi al proponente o per fissare degli incontri. Le principali motivazioni che hanno portato il team a selezionare Gmail* sono:

- La possibilità di utilizzare una firma personalizzata per le mail col fornitore e il committente in modo da uniformare e rendere più formali tutte le comunicazioni con soggetti esterni;
- La possibilità di utilizzare tutti gli strumenti di Google come Google Drive*, Google Meet*, Google Calendar* e altri.

4.2.3.1.4 Zoom*

Come strumento per i meeting esterni col fornitore e col committente il team ha deciso di utilizzare Zoom* che è una piattaforma per videoconferenze e riunioni online. Alcune delle sue caratteristiche sono:

- La possibilità di registrare le riunioni in modo da avere uno storico e risalire più facilmente a decisioni e scelte prese nelle fasi iniziali del progetto;
- La presenza di una chat e della possibilità di essere salvata al termine di ogni riunione in un formato di testo.

4.2.3.2 Pianificazione

In questa sottosezione verranno descritte più approfonditamente le tecnologie di pianificazione e gestione di progetto.

4.2.3.2.1 Jira*

Jira* è una suite di software per le gestione di progetti che mette a disposizione diverse funzionalità:

- La possibilità di creare ed assegnare task* ai vari membri del team;
- Una board che permette di monitorare i task* da svolgere, in corso e conclusi;
- Una roadmap che permette di mappare nel tempo i vari task*;
- Dei template per la stesura dei verbali.

4.3 Formazione

4.3.1 Scopo

Lo scopo della sezione è quello di definire le modalità e la pianificazione della formazione dei membri del team.

4.3.2 Aspettative

La motivazione del processo di formazione è quella di garantire un personale preparato adeguatamente in ogni fase del progetto rispetto agli strumenti e alle infrastrutture da utilizzare. Pianificare la formazione inoltre riduce il rischio di perdite di tempo causate da una mancata competenza in una specifica fase di progetto. La fase di formazione è costituita da uno studio autonomo da parte di ogni membro del gruppo di una specifica tecnologia o linguaggio.

4.3.3 Competenze

Le varie competenze per cui è richiesta una formazione si suddividono in diverse categorie in base ai vari aspetti del progetto.

4.3.3.1 Documentazione

Per la stesura dei documenti saranno richieste le seguenti competenze:

- **LaTeX***: è un linguaggio di markup che permette di sviluppare documenti con una resa grafica e una struttura migliore rispetto a strumenti come Google Docs* o Microsoft Word*;
- **Overleaf***: è un editor LaTeX* basato su cloud che permette di scrivere documenti in maniera collaborativa;
- **Jira***: per la stesura dei verbali durante i meeting verrà utilizzata la funzionalità di Jira che permette di avere dei documenti già strutturati suddivisi per sezione. In un secondo momento verrà poi prodotta una versione più formale in LaTeX*.

4.3.3.2 Organizzazione

Per l'organizzazione e la coordinazione del processo saranno richiesti i seguenti strumenti:

- **GitHub***: per la gestione del versionamento e della condivisione del progetto verrà utilizzato GitHub*. Oltre a garantire il lavoro collaborativo all'interno del team permette anche di gestire un sistema di branch* per rendere più ordinata la collaborazione (branch* per specifiche feature, branch* di sviluppo, branch* di release);
- **Jira***: per la gestione dei vari task* e per la loro suddivisione tra i membri del gruppo verrà utilizzata la suite di software Jira*. Questo strumento permette di utilizzare diversi strumenti utili per la gestione di progetto e per la pianificazione.

4.3.3.3 Sviluppo

Per la fase di sviluppo le librerie e i linguaggi richiesti sono i seguenti:

- **React***: libreria JavaScript* per lo sviluppo front end di interfacce utente;
- **JavaScript***: come richiesto dal committente per lo sviluppo verrà utilizzato anche il linguaggio JavaScript*;
- **HTML5* e CSS3***: per lo sviluppo dell'interfaccia front end sarà utilizzato HTML5* per la struttura e CSS3* per lo stile;
- **D3.js***: libreria JavaScript* per la rappresentazione dinamica dei dati attraverso grafici altamente personalizzabile;
- **Druid.js***: libreria JavaScript* che mette a disposizione gli algoritmi di riduzione che verranno utilizzati per il progetto.
- **Papa Parse***: libreria JavaScript* che permette di eseguire il parsing di file in formato CSV in JSON.