



# Norme di Progetto

## Informazioni sul Documento

<b>Versione</b>	0.0.1
<b>Approvatori</b>	
<b>Redattori</b>	Michele Gatto Pietro Villatora
<b>Verificatori</b>	Edoardo Pavan Francesco Protopapa Greta Cavedon Luciano Wu Matteo Basso
<b>Uso</b>	Interno
<b>Distribuzione</b>	Prof. Vardanega Tullio Prof. Cardin Riccardo Gruppo <i>DreamTeam</i>

e-mail: [dreamteam.unipd@gmail.com](mailto:dreamteam.unipd@gmail.com)

## Registro delle Modifiche

Versione	Data	Nominativo	Ruolo	Descrizione
v0.2.8	2022-01-21	Michele Gatto	Analista	Stesura §C, aggiunti §2.2.3.8, §2.2.4, §3.1.9, §3.4.6 e §4.1.4.3; (Verificatore: <i>Luciano Wu</i> )
v0.2.7	2022-01-20	Pietro Villatora	Analista	Stesura §B; (Verificatore: <i>Luciano Wu</i> )
v0.2.5	2022-01-16	Michele Gatto	Analista	Completata stesura §3.4 e §3.5; (Verificatore: <i>Greta Cavedon</i> )
v0.2.4	2022-01-15	Pietro Villatora	Analista	Stesura §A; (Verificatore: <i>Luciano Wu</i> )
v0.2.3	2022-01-13	Michele Gatto	Analista	Completata stesura §2.3 e §2.4; (Verificatore: <i>Greta Cavedon</i> )
v0.2.2	2022-01-10	Pietro Villatora	Analista	Completata stesura §2.1 e §2.2; (Verificatore: <i>Greta Cavedon</i> )
v0.2.1	2022-01-08	Pietro Villatora	Analista	Modifica di §3.1.7.5 e di §3.1.8.4; (Verificatore: <i>Francesco Protopapa</i> )
v0.2.0	2021-12-28	Michele Gatto	Analista	Aggiornata struttura cartelle come da §3.1.6, verifica complessiva di coesione e consistenza; (Verificatore: )
v0.1.5	2021-12-14	Pietro Villatora	Analista	Modifica di §3.1.7.2 e di §3.2.3; (Verificatore: <i>Francesco Protopapa</i> )
v0.1.4	2021-12-10	Michele Gatto	Analista	Completata stesura §3.2 e §3.3; (Verificatore: <i>Greta Cavedon</i> )
v0.1.3	2021-12-08	Michele Gatto	Analista	Completata stesura §3.1; (Verificatore: <i>Francesco Protopapa</i> )
v0.1.2	2021-12-06	Pietro Villatora	Analista	Inizio stesura §2; (Verificatore: <i>Greta Cavedon</i> )
v0.1.1	2021-12-04	Pietro Villatora	Analista	Inizio stesura §3; (Verificatore: <i>Francesco Protopapa</i> )
v0.1.0	2021-11-30	Pietro Villatora	Verificatore	Rivisitazione generale documento, verifica complessiva di coesione e consistenza. (Verificatore: <i>Greta Cavedon</i> )
v0.0.6	2021-11-29	Michele Gatto	Analista	Completata stesura §4.2; (Verificatore: <i>Matteo Basso</i> )
v0.0.5	2021-11-28	Michele Gatto	Analista	Completata stesura §4.1; (Verificatore: <i>Matteo Basso</i> )
v0.0.4	2021-11-27	Pietro Villatora	Analista	Inizio stesura §3; (Verificatore: <i>Edoardo Pavan</i> )
v0.0.3	2021-11-26	Pietro Villatora	Analista	Stesura §1 (Verificatore: <i>Edoardo Pavan</i> )



v0.0.2	2021-11-25	Michele Gatto	Analista	Inizio stesura §4 (Verificatore: <i>Matteo Basso</i> )
v0.0.1	2021-11-23	Pietro Villatora	Analista	Creazione scheletro documento; (Verificatore: <i>Michele Gatto</i> )

# Indice

<b>1</b>	<b>Introduzione</b>	<b>7</b>
1.1	Scopo del Documento . . . . .	7
1.2	Scopo del Prodotto . . . . .	7
1.3	Glossario . . . . .	7
1.4	Riferimenti . . . . .	7
1.4.1	Normativi . . . . .	7
1.4.2	Informativi . . . . .	7
<b>2</b>	<b>Processi Primari</b>	<b>8</b>
2.1	Fornitura . . . . .	8
2.1.1	Scopo . . . . .	8
2.1.2	Descrizione . . . . .	8
2.1.3	Rapporti con il proponente . . . . .	8
2.1.4	Materiale fornito . . . . .	8
2.2	Sviluppo . . . . .	8
2.2.1	Scopo . . . . .	8
2.2.2	Descrizione . . . . .	9
2.2.3	Analisi dei requisiti . . . . .	9
2.2.3.1	Scopo . . . . .	9
2.2.3.2	Descrizione . . . . .	9
2.2.3.3	Struttura . . . . .	9
2.2.3.4	Classificazione casi d'uso . . . . .	9
2.2.3.5	Modello di Diagramma di un Caso D'Uso . . . . .	10
2.2.3.6	Classificazione requisiti . . . . .	10
2.2.3.7	Qualità dei requisiti . . . . .	12
2.2.3.8	Metriche . . . . .	12
2.3	Progettazione . . . . .	12
2.3.1	Scopo . . . . .	12
2.3.2	Descrizione . . . . .	12
2.3.3	Technology Baseline . . . . .	12
2.3.4	Product Baseline . . . . .	12
2.3.5	Qualità . . . . .	13
2.4	Codifica . . . . .	13
2.4.1	Scopo . . . . .	13
2.4.2	Descrizione . . . . .	13
2.4.3	Stile di codifica . . . . .	13
2.4.3.1	Norme di buona programmazione . . . . .	13
2.4.3.2	Norme di struttura . . . . .	13
2.4.3.3	Convenzioni sulla nomenclatura . . . . .	13
2.4.3.4	Brevità dei metodi . . . . .	14
2.4.4	Metriche . . . . .	14
2.4.5	Strumenti . . . . .	14
<b>3</b>	<b>Processi di Supporto</b>	<b>15</b>
3.1	Documentazione . . . . .	15
3.1.1	Scopo . . . . .	15
3.1.2	Descrizione . . . . .	15
3.1.3	Documenti prodotti . . . . .	15
3.1.4	Sistema software per la preparazione dei documenti . . . . .	15
3.1.5	Ciclo di vita di un documento . . . . .	15
3.1.6	Struttura delle directory e dei files . . . . .	16
3.1.7	Struttura di un documento . . . . .	16
3.1.7.1	Prima pagina . . . . .	16
3.1.7.2	Registro delle modifiche . . . . .	17
3.1.7.3	Indice . . . . .	17
3.1.7.4	Struttura delle pagine . . . . .	17
3.1.7.5	Verbali . . . . .	17



3.1.8	Normativa tipografica . . . . .	18
3.1.8.1	Nomi dei documenti . . . . .	18
3.1.8.2	Stile di testo . . . . .	18
3.1.8.3	Termini di glossario . . . . .	18
3.1.8.4	Elementi testuali . . . . .	18
3.1.8.5	Elementi grafici . . . . .	19
3.1.9	Metriche . . . . .	19
3.1.10	Strumenti . . . . .	19
3.2	Gestione della configurazione . . . . .	19
3.2.1	Scopo . . . . .	19
3.2.2	Descrizione . . . . .	20
3.2.3	Versionamento . . . . .	20
3.2.3.1	Strumenti . . . . .	20
3.2.4	Struttura del repository . . . . .	20
3.2.5	Modifiche al repository . . . . .	20
3.3	Gestione della qualità . . . . .	20
3.3.1	Scopo . . . . .	20
3.3.2	Descrizione . . . . .	20
3.3.3	Attività di processo . . . . .	21
3.3.4	Controllo di qualità . . . . .	21
3.3.5	Denominazione metriche e obiettivi . . . . .	21
3.4	Verifica . . . . .	21
3.4.1	Scopo . . . . .	21
3.4.2	Descrizione . . . . .	21
3.4.3	Verifica della documentazione . . . . .	22
3.4.3.1	Walkthrough . . . . .	22
3.4.3.2	Inspection . . . . .	22
3.4.4	Verifica del codice . . . . .	22
3.4.4.1	Test . . . . .	22
3.4.4.2	Nominazione dei test . . . . .	22
3.4.5	Verifica dei requisiti . . . . .	23
3.4.6	Metriche . . . . .	23
3.5	Validazione . . . . .	23
3.5.1	Scopo . . . . .	23
3.5.2	Descrizione . . . . .	23
<b>4</b>	<b>Processi Organizzativi</b>	<b>24</b>
4.1	Gestione di processo . . . . .	24
4.1.1	Scopo . . . . .	24
4.1.2	Obiettivi . . . . .	24
4.1.3	Coordinamento . . . . .	24
4.1.3.1	Comunicazione . . . . .	24
4.1.3.2	Riunioni . . . . .	25
4.1.4	Pianificazione . . . . .	25
4.1.4.1	Ruoli di progetto . . . . .	25
4.1.4.2	Gestione dei ticket . . . . .	27
4.1.4.3	Metriche . . . . .	28
4.2	Formazione dei membri del team . . . . .	28
4.2.1	Obiettivi . . . . .	28
4.2.2	Formazione interna . . . . .	28
4.2.3	Formazione esterna . . . . .	28
4.2.4	Strumenti a supporto della gestione organizzativa . . . . .	29
<b>A</b>	<b>Standard ISO/IEC 15504 - SPICE</b>	<b>30</b>
A.1	Introduzione . . . . .	30
A.1.1	Classificazione processi . . . . .	30
A.2	Livelli di capability e attributi di processo . . . . .	30



<b>B</b>	<b>Metriche per la qualità del processo</b>	<b>32</b>
B.1	MPC01 SPICE . . . . .	32
B.2	MPC02 Budgeted cost of work scheduled . . . . .	32
B.3	MPC03 Actual cost of work performed . . . . .	32
B.4	MPC04 Budgeted cost of work performed . . . . .	32
B.5	MPC05 Schedule variance . . . . .	32
B.6	MPC06 Budget variance . . . . .	32
B.7	MPC07: Requirements stability . . . . .	32
<b>C</b>	<b>Metriche per la qualità del processo</b>	<b>33</b>
C.1	MQP01 Indice di Gulpease . . . . .	33
C.2	MQP02 Profondità di una gerarchia . . . . .	33
C.3	MQP03 Numero parametri per metodo . . . . .	33
C.4	MQP04 Code coverage . . . . .	33
C.5	MQP05 Percentuale requisiti obbligatori soddisfatti . . . . .	33
C.6	MQP06 Complessità ciclomatica . . . . .	33
C.7	MQP07 Numero di bug . . . . .	34
C.8	MQP08 Numero di code smell . . . . .	34
C.9	MQP09 Linee di Commento per Linee di Codice . . . . .	34
C.10	MQP10 Branch coverage . . . . .	34
C.11	MQP11 Successo dei test . . . . .	34
C.12	MQP12 Numero di bug . . . . .	34

## Elenco delle figure

1	Esempio di Caso D'Uso . . . . .	10
2	Attività di gestione dei ticket . . . . .	27



# 1 Introduzione

## 1.1 Scopo del Documento

Lo scopo di questo documento è di definire le norme, le convenzioni e le procedure adottate da tutti i membri di *DreamTeam*, in modo da poter definire un metodo di lavoro comune. Per raggiungere questo scopo ogni membro è tenuto a visionare periodicamente il documento e a rispettare tutte le norme in esso presenti. Per la redazione viene adottata una filosofia incrementale, quindi il documento allo stato attuale è incompleto e le norme saranno definite passo passo partendo dalle più urgenti, con l'aspettativa di avere un processo normato prima del suo avvio, considerando che, in generale, ogni norma può essere soggetta a cambiamenti.

## 1.2 Scopo del Prodotto

L'obiettivo di Sweeat dell'azienda Zero12 è la creazione di un sistema software costituito da una Webapp. Lo scopo del prodotto è di fornire all'utente una guida dei locali gastronomici sfruttando i numerosi contenuti digitali creati dagli utenti sulle principali piattaforme social (Instagram e TikTok). In questo modo, è possibile realizzare una classifica basata sulle impressioni e reazioni di chiunque usufruisca dei servizi dei locali, non solo da professionisti ed esperti del settore.

## 1.3 Glossario

Per evitare ambiguità relative alle terminologie utilizzate è stato creato un documento denominato “*Glossario*”. Questo documento comprende tutti i termini tecnici scelti dai membri del gruppo e utilizzati nei vari documenti con le relative definizioni. Tutti i termini inclusi in questo glossario, vengono segnalati all'interno del documento con l'apice <sup>G</sup> accanto alla parola.

## 1.4 Riferimenti

### 1.4.1 Normativi

- Presentazione del capitolato - Zero12 Progettazione e sviluppo di una Social guida Michelin: <https://www.math.unipd.it/~tullio/IS-1/2021/Progetto/C4.pdf>;

### 1.4.2 Informativi

- Standard ISO/IEC 9126: [https://it.wikipedia.org/wiki/ISO/IEC\\_9126](https://it.wikipedia.org/wiki/ISO/IEC_9126);
- Standard ISO/IEC 15504: [https://en.wikipedia.org/wiki/ISO/IEC\\_15504](https://en.wikipedia.org/wiki/ISO/IEC_15504).

## 2 Processi Primari

### 2.1 Fornitura

#### 2.1.1 Scopo

Lo scopo del processo di fornitura, come definito nello standard ISO/IEC/IEEE 12207:1995, è di determinare quali strumenti, risorse e competenze siano necessarie per lo svolgimento del progetto.

#### 2.1.2 Descrizione

La presente sezione contiene tutte le norme che ogni membro del gruppo è tenuto a seguire, durante le varie fasi di svolgimento del progetto, per poter divenire fornitori del proponente zero12 e dei committenti Prof. Tullio Vardanega e Prof. Riccardo Cardin. Verrà definita la gestione dei rapporti con il proponente, comprese consegna e manutenzione del prodotto finale.

#### 2.1.3 Rapporti con il proponente

Il gruppo entrerà in contatto con il proponente zero12 per:

- Approfondire aspetti chiave con il proponente per far fronte ai suoi bisogni;
- Chiarire ogni eventuale dubbio emerso;
- Stimare tempistiche di lavoro;
- Individuare le strategie lavorative più efficaci;
- Definizione dei requisiti e vincoli da rispettare;
- Proporre nuove soluzioni o alternative, discutendo di vantaggi, svantaggi e fattibilità.

A meno di accordi diversi con il proponente non sarà prevista nessuna manutenzione futura del prodotto dopo l'avvenuta consegna e collaudo.

#### 2.1.4 Materiale fornito

Il materiale che il gruppo fornirà al proponente e ai committenti sono:

- **Analisi dei Requisiti v1.0.0** contiene l'analisi dei casi d'uso e dei requisiti con lo scopo di:
  - Determinare tutte e sole le funzionalità che saranno offerte dal prodotto finale;
  - Chiarire ogni ambiguità che potrebbe sorgere nella comprensione del capitolato.
- **Piano di Progetto v1.0.0** contiene la pianificazione preventiva dei tempi, l'analisi dei rischi, il consuntivo di periodo, la data di consegna e i costi previsti;
- **Piano di Qualifica v1.0.0** contiene le modalità adottate in verifica e validazione, assicurando che la qualità dei processi e dei prodotti rispetti le aspettative.
- **Proof of Concept** piccolo software di esempio che servirà al gruppo per determinare la fattibilità pratica e dimostrare la fondatezza e applicabilità di concetti fondamentali e costituenti in relazione al prodotto finale.

## 2.2 Sviluppo

### 2.2.1 Scopo

Lo scopo del processo di sviluppo, secondo lo standard ISO/IEC/IEEE 12207:1995 è definire compiti e attività da eseguire per realizzare il prodotto finale richiesto dal proponente.



### 2.2.2 Descrizione

Sono elencate e dopo trattate le seguenti attività di questo processo:

- Analisi dei requisiti;
- Progettazione;
- Codifica.

### 2.2.3 Analisi dei requisiti

#### 2.2.3.1 Scopo

É compito di ogni *Analista* scrivere il documento di *Analisi dei Requisiti*. Lo scopo di tale documento è:

- aiutare i *Progettisti*;
- stabilire ciò che si è concordato con il cliente;
- fornire una base per chiunque prenda sottomano il prodotto per miglioramenti;
- aiutare le revisioni del codice;
- fornire riferimenti utili ai *Verificatori*;
- tracciare il lavoro per stimarne I costi.

#### 2.2.3.2 Descrizione

L'obiettivo è la realizzazione dell'architettura del sistema.

#### 2.2.3.3 Struttura

La struttura potrà essere soggetta a cambiamenti. Attualmente, *Analisi dei Requisiti v1.0.0* presenta questa struttura:

- Introduzione al documento;
- Descrizione generale, dove sono presenti requisiti estrapolati sia dal capitolato d'appalto che dagli incontri effettuati con il proponente (verbali esterni);
- Casi d'uso relativi alle varie funzionalità fruibili nella piattaforma<sup>G</sup>;
- Requisiti che dovrà soddisfare la piattaforma.

#### 2.2.3.4 Classificazione casi d'uso

La struttura adottata per la classificazione dei casi d'uso è la seguente:

**UC[Identificatore][CodiceCasoBase](.[CodiceSottoCaso])\***

Composta da:

- **UC**: acronimo di "Use Case";
- **Identificatore**: che può assumere le diverse espressioni letterali:
  - **W**: identifica un caso d'uso relativo alla WebApp;
  - **E**: identifica un caso d'uso d'errore.
- **CodiceCasoBase**: ID del caso d'uso generico;
- **CodiceSottoCaso**: ID opzionale per i sottocasi di un caso d'uso.

Ogni caso d'uso è descritto da:

- **Id**: codice identificativo del caso d'uso, stabilito come enunciato sopra;
- **Nome**: stringa titolo del caso d'uso posta dopo l'id;

- **Diagramma UML:** diagramma per rappresentare graficamente il caso d'uso;
- **Descrizione:** breve descrizione del caso d'uso;
- **Attori:** entità esterne al sistema che interagiscono con esso. Ne esistono due varianti :
  - **Primario:** interagisce con il sistema per raggiungere un obiettivo;
  - **Secondario:** aiuta il primario a raggiungere l'obiettivo. Non utilizzato.
- **Precondizione:** descrive lo stato del sistema prima del verificarsi del caso d'uso;
- **Postcondizione:** descrive lo stato del sistema dopo che si è verificato il caso d'uso;
- **Scenario principale:** elenco numerato che descrive il flusso degli eventi del caso d'uso;
- **Scenario secondario/alternativo :** elenco numerato che descrive il flusso degli eventi del caso d'uso dopo un evento imprevisto che lo ha deviato dal caso principale. Può non esserci o possono esserci più di uno;
- **Estensioni:** utilizzate nei scenari alternativi. Se si verifica una determinata situazione, il caso d'uso collegato all'estensione viene interrotto.

### 2.2.3.5 Modello di Diagramma di un Caso D'Uso

Qui di seguito viene mostrato un esempio di diagramma di un caso d'uso con tutte le componenti UML usate.

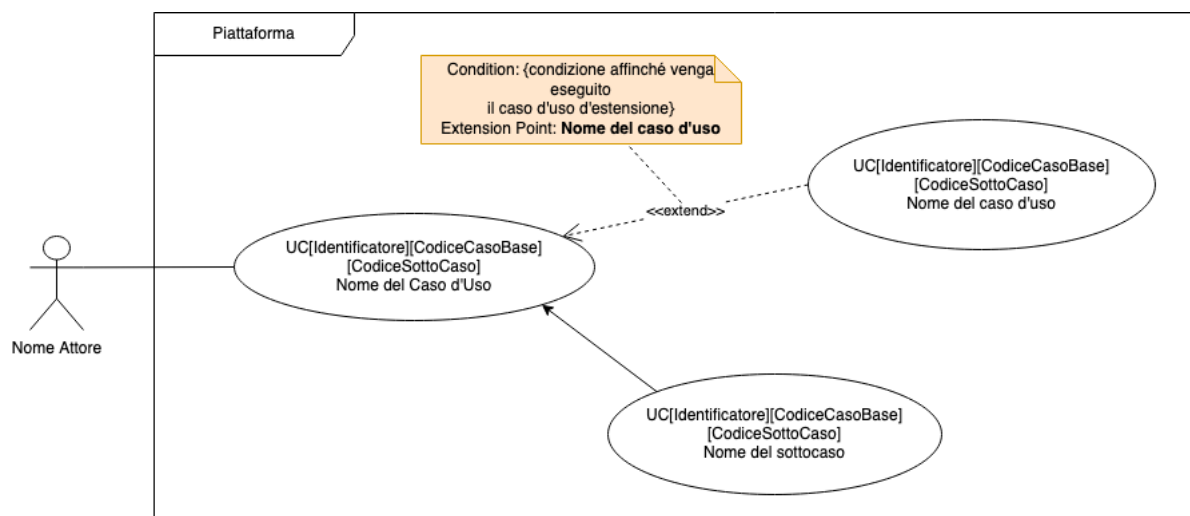


Figura 1: Esempio di Caso D'Uso

### 2.2.3.6 Classificazione requisiti

Un requisito è un obiettivo, accordato con il proponente o preso mediante una decisione interna, da raggiungere per risolvere un determinato problema. Per rappresentare un requisito, è stato scelto di adottare la seguente convenzione:

**R[Importanza][Tipologia][Codice]**

dove:

- **Importanza:** rappresenta l'importanza associata al requisito e può assumere uno dei seguenti valori:
  - **1:** Requisito *Obbligatorio*, la sua soddisfazione dovrà necessariamente avvenire per garantire una buona funzionalità dell'intero sistema;

- **2:** Requisito *Desiderabile*, la sua soddisfazione non vincola il buon funzionamento del sistema, tuttavia ne fornisce una maggior completezza;
- **3:** Requisito *Facoltativo*, se soddisfatto rende il sistema più completo, ma ciò potrebbe comportare un dispendio di energie con un conseguente aumento dei costi preventivati.
- **Tipologia:** si riferisce alla tipologia di requisito e può assumere uno dei seguenti valori letterali:
  - **V:** requisito di *Vincolo*, descrive i vincoli offerti dal sistema;
  - **F:** requisito *Funzionale*, descrive servizi o funzioni offerti dal sistema;
  - **P:** requisito *Prestazionale*, descrive i vincoli sulle prestazioni da soddisfare, con il numero di informazioni da manipolare in un certo intervallo di tempo;
  - **Q:** requisito di *Qualità*, descrive i vincoli di qualità da realizzare (xxx).
- **Codice:** identifica in maniera univoca il requisito in forma gerarchica padre/figlio. Per esplicitare la forma gerarchica, il codice viene rappresentato come segue:

**[Identificatore][CodiceBase](.[CodiceSottoCaso])**

dove:

- **Identificatore:** può essere presente o meno. Nel caso sia presente, può assumere una delle seguenti espressioni letterali:
  - \* **W:** identifica un caso d'uso relativo alla WebApp;
  - \* **E:** identifica un caso d'uso di un errore.
- **CodiceBase:** fa riferimento al caso d'uso preso in esame e, in combinazione con la Tipologia, definisce un identificatore univoco per il requisito;
- **CodiceSottoCaso:** codice progressivo opzionale, che può includere più livelli, ed identifica un eventuale sottocaso.

Dopo aver classificato ciascun requisito con un codice, quest'ultimo non potrà più essere cambiato. Inoltre, ciascun codice verrà accompagnato da una serie di informazioni aggiuntive, che meglio definiranno ciascun requisito, ossia:

- **Descrizione:** breve descrizione completa relativa allo scopo del requisito;
- **Classificazione:** indica l'importanza del requisito e può assumere i valori *Obbligatorio*, *Desiderabile* e *Facoltativo*. Sebbene questa informazione possa sembrare ridondante, ne facilita la lettura;
- **Fonti:** indica le fonti del requisito, ossia possono essere:
  - all'interno del Capitolato d'Appalto,
  - nei Verbali Interni,
  - nei Verbali Esterni
  - nei Casi d'Uso presenti nel documento “*Analisi dei Requisiti v1.0.0*”
  - una decisione presa internamente (quindi “Decisione Interna”).

Ad esempio:

Requisito	Descrizione	Classificazione	Fonti
R1FW1	L'utente deve riuscire ad inserire i propri dati personali (nome, cognome, indirizzo e-mail e password) per effettuare la registrazione	Obbligatorio	UCW1

### 2.2.3.7 Qualità dei requisiti

Ciascun requisito deve essere:

- Completo, ovvero dettagliato;
- Consistente, che non sia in contraddizione con altri requisiti;
- Necessario;
- Verificabile, ovvero che sia possibile controllare che il sistema lo realizzi.
- Tracciabile.

### 2.2.3.8 Metriche

Le metriche utilizzate per la valutazione dell'attività di analisi dei requisiti sono:

- **MQP05 Percentuale requisiti obbligatori soddisfatti** (per maggiori informazioni consultare l'appendice C.5);
- **MPC07: Requirements stability** (per maggiori informazioni consultare l'appendice B.7).

## 2.3 Progettazione

### 2.3.1 Scopo

Lo scopo della Progettazione è di determinare le caratteristiche che il prodotto deve avere per soddisfare i requisiti individuati dagli stakeholder. Il procedimento seguito è l'opposto rispetto a quello usato per l'*Analisi dei Requisiti v1.0.0*. Vengono individuate le diverse parti, coerenti con i requisiti, che verranno poi raggruppate in vari sottoinsiemi fino ad arrivare ad un'unica soluzione finale. Restano da rispettare i vincoli di sostenibilità nell'utilizzo delle risorse, nel contenimento dei costi e nel rispetto degli obiettivi di qualità.

### 2.3.2 Descrizione

L'obiettivo è la realizzazione dell'architettura del sistema.

### 2.3.3 Technology Baseline

Misura la comprensione delle tecnologie individuate per la realizzazione del prodotto, motivandone la scelta. Dovranno essere mostrate:

- Le varie tecnologie adottate, motivando le scelte;
- Le relazioni tra i vari componenti e come interagiscono tra di loro;
- Il ***Proof of Concept***: un prototipo (incompleto) eseguibile che riunisce tutte le tecnologie adottate, dimostrando in maniera pratica la loro adeguatezza e compatibilità reciproca.

### 2.3.4 Product Baseline

Rappresenta la baseline architetturale (design e coding) del prodotto, coerente con la Technology Baseline. Mostra il design definitivo del prodotto. Dovrà contenere:

- Diagrammi UML delle classi e di sequenza;
- Tracciamento per ogni classe dei requisiti che deve soddisfare;
- Descrizione dei design pattern utilizzati.

### 2.3.5 Qualità

È compito del *Progettista* definire un'architettura di qualità. Le caratteristiche che essa dovrà avere sono :

- Soddisfare i requisiti indicati nel documento *Analisi dei Requisiti v1.0.0*;
- Essere comprensibile, robusta e affidabile;
- Presentare componenti semplici, in maniera tale da garantire modularità e riusabilità, semplificando il lavoro dello sviluppatore;
- Utilizzare le risorse in maniera efficiente.

*Questa sezione sarà soggetta a modifiche future con l'avanzare del progetto.*

## 2.4 Codifica

### 2.4.1 Scopo

Lo scopo del processo di codifica è l'effettiva realizzazione del prodotto software, svolto dal *Programmatore*. Si può vedere come la trasformazione in codice dell'architettura definita dai *Progettisti*.

### 2.4.2 Descrizione

Il codice deve rispettare gli obiettivi di qualità definiti nel *Piano di Qualifica v1.0.0*. Nelle sezioni sottostanti saranno elencate regole e norme di carattere più generale, utilizzate da ogni linguaggio di programmazione impiegato nel progetto.

### 2.4.3 Stile di codifica

#### 2.4.3.1 Norme di buona programmazione

- Produrre codice leggibile e verificabile;
- Strutturare la codifica in modo da rispettare il design progettato;
- Massimizzare l'information hiding.

#### 2.4.3.2 Norme di struttura

- **Indentazione:** i blocchi di codice innestati dovranno avere un'indentazione di quattro spazi, evitando l'utilizzo di tabulazioni;
- **Parentesi:** le parentesi aperte di delimitazione dei blocchi di codice dovranno trovarsi in linea con l'istruzione che definisce il blocco di codice. Eccezione ovvia la fanno i linguaggi che non usano questo sistema di delimitazione dei blocchi di codice;
- **Commenti:** i commenti al codice saranno presentati in lingua italiana. Se necessari dovranno trovarsi generalmente sopra al costrutto in questione;
- **Lunghezza riga:** una riga di codice dovrà essere lunga al massimo 140 caratteri, altrimenti dovrà essere spezzata in maniera uniforme;
- **Univocità dei nomi:** tutti i costrutti dovranno avere nomi univoci ed autoesplicativi, da evitare nomi eccessivamente lunghi.

#### 2.4.3.3 Convenzioni sulla nomenclatura

In generale per i nomi verrà utilizzata la nomenclatura a cammello (camel case) con alcune particolarità:

- Nomi di variabili e metodi dovranno cominciare con la lettera minuscola;
- Nomi di classi, file e cartelle dovranno cominciare con la lettera maiuscola;
- Nomi di costanti dovranno essere scritti tutti in maiuscolo, separando le parole con il carattere underscore " \_ " .



#### 2.4.3.4 Brevità dei metodi

In generale ogni metodo o procedura dovrebbe essere breve e conciso, tra le 20 e 25 righe (contando un'istruzione per riga). Poiché il bisogno di scrivere un metodo più lungo può diventare necessario, si possono considerare tollerabili procedure fino ad un massimo di 35-40 righe. Nel caso in cui la soglia venga superata, ha senso considerare una suddivisione della procedura. È fondamentale concentrarsi sullo scopo di ogni procedura, evitando singole procedure che svolgono troppi compiti.

#### 2.4.4 Metriche

Per la valutazione del processo di codifica saranno adottate le seguenti metriche:

- **MQP02 Profondità di una gerarchia;**
- **MQP03 Numero parametri per metodo;**
- **MQP05 Percentuale requisiti obbligatori soddisfatti;**
- **MQP06 Complessità ciclomatica;**
- **MQP07 Numero di bug;**
- **MQP08 Numero di code smell;**
- **MQP09 Linee di Commento per Linee di Codice;**
- **MQP12 Numero di vulnerabilità.**

Per maggiori informazioni consultare §C.

#### 2.4.5 Strumenti

Gli strumenti che verranno adottati durante il processo di sviluppo sono:

- **Visual Studio Code:** IDE utilizzato dal gruppo per la stesura del codice;
- **AWS CLI:** uno strumento unificato per la gestione dei servizi AWS <sup>G</sup> via interfaccia a riga di comando.

## 3 Processi di Supporto

### 3.1 Documentazione

#### 3.1.1 Scopo

Ogni processo e attività per lo sviluppo del progetto dovrà essere documentata. Nella presente sezione verranno descritte regole e standard da seguire durante il processo di documentazione per l'intero ciclo di vita del software.

#### 3.1.2 Descrizione

Vengono presentate decisioni e norme prescelte per:

- Stesura;
- Verifica;
- Approvazione.

#### 3.1.3 Documenti prodotti

I documenti prodotti sono:

- **Norme di progetto:** documento interno che contiene norme e regole stabilite dal gruppo, che devono essere seguite per l'intera durata del progetto;
- **Glossario:** documento esterno dove sono presenti i termini tecnici usati nella documentazione con le loro definizioni, affinché non ci siano ambiguità e/o incongruenze;
- **Piano di progetto:** documento esterno con la pianificazione delle attività del progetto previste dal gruppo. Contiene la previsione dell'impegno orario dei singoli membri, il preventivo spese e i consuntivi di periodo;
- **Piano di qualifica:** documento esterno che descrive i criteri con cui si valuta la qualità;
- **Analisi dei requisiti:** documento esterno contenente requisiti e caratteristiche del prodotto finale;
- **Verbali:**
  - Interni: resoconti degli incontri del gruppo;
  - Esterni: resoconti degli incontri del gruppo con i committenti e/o il proponente.

#### 3.1.4 Sistema software per la preparazione dei documenti

Tutti i documenti prodotti dal gruppo verranno redatti usando il linguaggio di markup  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ .

#### 3.1.5 Ciclo di vita di un documento

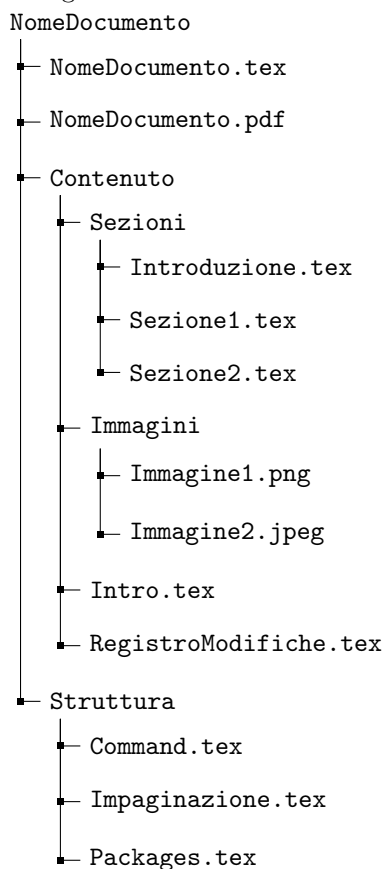
Ogni documento passa per i seguenti step:

- **Creazione:** il documento viene creato basandosi su un template comune;
- **Strutturazione:** il documento viene fornito di:
  - Registro delle modifiche;
  - Indice dei contenuti.
- **Stesura:** il gruppo redige il documento adottando il metodo incrementale;
- **Revisione:** ogni sezione del corpo del documento è rivista da almeno un membro del gruppo che non sia il redattore della parte in verifica;
- **Approvazione:** se revisionato, il Responsabile di Progetto può stabilire che il documento è valido. Se approvato, può essere rilasciato.

Per semplificare le operazioni di verifica dovrà sempre essere reso disponibile il documento completo (fino alla versione più attuale) in formato PDF.

### 3.1.6 Struttura delle directory e dei files

Per ogni documento si definisce la seguente struttura di directory:



In particolare:

- **NomeDocumento.tex**: importa tutti le parti necessarie per comporre il documento finale;
- **NomeDocumento.pdf**: versione del documento in formato PDF;
- **Introduzione.tex**: sezione introduttiva del documento, definisce lo scopo del prodotto e del documento, i riferimenti normativi e informativi e informazioni sul *Glossario*;
- **Intro.tex**: contiene tutti i comandi per creare la prima pagina del documento;
- **RegistroModifiche.tex**: contiene tutti i comandi per creare la tabella del registro delle modifiche;
- **Command.tex**: contiene tutti i comandi aggiuntivi creati dal gruppo;
- **Impaginazione.tex**: definisce alcune istruzioni per l'impaginazione e si occupa di creare header e footer del documento;
- **Packages.tex**: contiene tutti i pacchetti aggiuntivi e necessari per la compilazione.

### 3.1.7 Struttura di un documento

#### 3.1.7.1 Prima pagina

La prima pagina è composta da:

- Logo del gruppo;
- Titolo del documento;
- Informazioni varie del documento:



- **Versione corrente;**
- **Approvatori:** indica chi ha approvato il documento. Se non presente, indica che il documento non è ancora stato approvato;
- **Data approvazione;**
- **Redattori:** indica chi si è occupato della stesura del documento;
- **Verificatori:** indica chi si è occupato della verifica del documento;
- **Uso:** indica se il documento è dedicato a uso interno o esterno;
- **Distribuzione:** indica a chi viene distribuito il documento;

- **Indirizzo e-mail del gruppo.**

### 3.1.7.2 Registro delle modifiche

Ogni documento ha il suo registro modifiche che tiene traccia di tutte le modifiche importanti del documento durante il suo ciclo di vita. Sotto forma di tabella, riporta:

- Versione del documento dopo la modifica;
- Data della modifica;
- Nome dell'autore della modifica;
- Ruolo dell'autore al momento della modifica;
- Descrizione breve della modifica;
- Nome della persona che si è occupata di verificare la modifica.

Inoltre, nella descrizione di ciascuna modifica viene indicato anche il paragrafo interessato con questa convenzione:

- §: indica un solo paragrafo,
- §§: indica l'intervallo di paragrafi su cui sono state apportate le modifiche fatte.

### 3.1.7.3 Indice

Presente dopo il registro delle modifiche, l'indice permette di avere una visione completa del documento e di individuare le varie parti, ogni voce è un collegamento ipertestuale alla parte del documento in cui viene trattata.

### 3.1.7.4 Struttura delle pagine

Ogni pagina, a eccezione della prima, è formata da questi elementi:

- In alto a sinistra si trova una miniatura a colori del logo del gruppo;
- In alto a destra è presente il titolo del documento;
- Sotto i due elementi appena elencati una linea nera continua li separa dal contenuto della pagina;
- Il contenuto della pagina;
- Sul lato destro del piè di pagina è indicato il numero della pagina corrente;

### 3.1.7.5 Verballi

I verballi applicano le stesse norme strutturali degli altri documenti con la differenza che non sono soggetti a versionamento. Ogni verbale sia interno che esterno dovrà contenere:

- Motivo della riunione;
- Luogo della riunione;
- Data della riunione;

- Orario di inizio e fine riunione;
- Partecipanti della riunione;
- Resoconto della riunione;
- Registro delle decisioni, dove si riporta in tabella le decisioni prese dal gruppo durante l'incontro;

### 3.1.8 Normativa tipografica

#### 3.1.8.1 Nomi dei documenti

La struttura generale del nome è la seguente:

[NomeDocumento]-v[X].[Y].[Z]

in particolare:

- [NomeDocumento] inizia sempre con la lettera maiuscola. Se presenti più parole, queste saranno attaccate ma distinguibili dalla lettera maiuscola (convenzione "*CamelCase*");
- v[X].[Y].[Z] rappresenta la versione corrente del documento seguendo lo schema di versionamento presentato in TODO: metti i link;

I verbali, in quanto non soggetti a versionamento avranno una struttura del nome diversa, ovvero:

Verbale[Tipologia]-[YYYY].[MM].[DD]

dove:

- [Tipologia] intende il tipo del verbale, può essere **Interno** o **Esterno**;
- [YYYY].[MM].[DD] indica la data in cui è avvenuto l'incontro.

#### 3.1.8.2 Stile di testo

Gli stili di testi adottati nei documenti sono:

- **Grassetto**: per titoli, sottotitoli, e altri termini ritenuti importanti dal redattore;
- **Maiuscolo**: per acronimi e iniziali di nomi propri, dei documenti o dei paragrafi;
- **Corsivo**: per nomi propri dei membri del gruppo, committenti e proponente e per i nomi dei documenti.

#### 3.1.8.3 Termini di glossario

I termini che possono risultare ambigui e/o incongruenti sono contrassegnati con una <sup>G</sup> alla loro prima occorrenza nella sezione d'interesse. Questi termini sono riportati con il loro significato in un documento esterno, il *Glossario*.

#### 3.1.8.4 Elementi testuali

I redattori devono seguire le seguenti regole stilistiche:

- **Elenchi puntati**: un elenco puntato utilizzerà il simbolo • (pallino). Un successivo annidamento utilizzerà il simbolo - (trattino) e un altro ancora un asterisco (\*). Se si tratta di un elenco numerato, i quattro livelli di enumerazione sono ordinati con i numeri arabi divisi da un punto fermo. Ogni voce dell'elenco inizia con una lettera maiuscola e termina con un punto e virgola, tranne l'ultima voce che termina con un punto;
- **Formati di data**: Le date usano il formato [YYYY]-[MM]-[DD] dove:
  - [YYYY] corrisponde all'anno;
  - [MM] corrisponde al mese;
  - [DD] corrisponde al giorno.

- **Orario:** gli orari usano il formato **[HH]:[MM]** dove:
  - **[HH]** rappresentano le ore;
  - **[MM]** rappresentano i minuti.
- **Sigle:** Tutte le sigle hanno le iniziali di ogni parola maiuscola tranne preposizioni, congiunzioni e articoli. Le sigle utilizzate sono:
  - Relative ai documenti:
    - \* **Analisi dei Requisiti:** AdR;
    - \* **Piano di Progetto:** PdP;
    - \* **Piano di Qualifica:** PdQ;
    - \* **Glossario:** G;
    - \* **Norme di Progetto:** NdP;
    - \* **Verbali Interni:** VI;
    - \* **Verbali Esterni:** VE.
  - Relative ai ruoli di progetto:
    - \* **Responsabile di Progetto:** RE;
    - \* **Amministratore:** AM;
    - \* **Analista:** AN;
    - \* **Progettista:** PT;
    - \* **Programmatore:** PR;
    - \* **Verificatore:** VE.

#### 3.1.8.5 Elementi grafici

Le regole per quanto riguarda l'uso di elementi grafici sono:

- **Immagini:** le figure presenti sono centrate rispetto al testo e accompagnate da didascalia;
- **Diagrammi UML:** verranno inseriti nel documento tramite delle immagini

#### 3.1.9 Metriche

Per la valutazione dei documenti prodotti del processo di documentazione, verranno applicate la seguente metrica di prodotto:

- **MQP01 Indice di Gulpease**, trattato nell'appendice C.1.

#### 3.1.10 Strumenti

Gli strumenti dedicati alla stesura sono:

- **L<sup>A</sup>T<sub>E</sub>X**: linguaggio compilato basato sul programma di composizione tipografica Tex;
- **Texmaker**: l'editor per la stesura dei documenti;
- **Overleaf**: editor per la stesura dei documenti basato sul cloud;
- **Draw.io**: utilizzato per la creazione di grafici UML.

### 3.2 Gestione della configurazione

#### 3.2.1 Scopo

Lo scopo è di gestire e controllare la produzione di documenti e codice in maniera sistematica. Per ogni oggetto sottoposto a configurazione viene garantito il versionamento e controllo sulle modifiche per permettere il mantenimento dell'integrità del prodotto.

### 3.2.2 Descrizione

Vengono raggruppati e organizzati tutti i mezzi usati per la configurazione degli strumenti designati alla produzione di documenti e codice, per poter gestire struttura e la disposizione dei file all'interno di repository e anche quelli per versionamento e coordinamento.

### 3.2.3 Versionamento

Per poter capire lo stato di avanzamento di un prodotto delle attività del progetto è necessario un identificatore. Il formato del codice di versione utilizzato è

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

dove :

- **X** indica il rilascio pubblico e corrisponde ad una versione approvata dal Responsabile di Progetto. La numerazione parte da 0;
- **Y** indica una revisione complessiva del prodotto per verificare che, dopo una modifica, il prodotto sia ancora coeso e consistente. La numerazione parte da 0 e si azzerà ad ogni incremento di X;
- **Z** viene incrementato ad ogni modifica con relativa verifica. La numerazione parte da 0 e si azzerà ad ogni incremento di X o Y.

#### 3.2.3.1 Strumenti

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

### 3.2.4 Struttura del repository

Il repository utilizzato dal gruppo per la creazione dei documenti contiene una directory per ogni documento denominata **NomeDocumento**, la cui struttura è approfondita in TODO: metti i link. Il repository è suddiviso in più branch così definiti:

- **main**: il branch principale, che contiene l'ultima versione verificata di ogni documento;
- **NomeDocumento**: uno per ogni documento, è dove il documento vive e viene attivamente stilato dai membri del gruppo.

### 3.2.5 Modifiche al repository

Non è consentito fare commit direttamente sul branch **main**, poiché porterebbe ad un elevato rischio di incongruenze e merge conflicts. Potrà essere modificato solo tramite il meccanismo di pull request con verifica obbligatoria, in modo da garantire che sia sempre presente una versione verificata e corretta del documento, anche se incompleta. Nel branch **NomeDocumento** invece ogni membro può fare commit a patto che siano relative solo al documento specificato. Ogni commit deve referenziare la issue da cui è derivata e quindi, in generale, potranno effettuare modifiche sul branch solo gli assegnatari di issue che trattano quel documento specifico. Per quanto riguarda cambiamenti minimali (punteggiatura, errori ortografici, ecc.) è permessa la modifica autonoma da parte di qualsiasi membro del gruppo e non è necessario referenziare nessuna issue.

## 3.3 Gestione della qualità

### 3.3.1 Scopo

Lo scopo del processo di gestione della qualità è di assicurare che i requisiti di qualità individuati dagli stakeholder e le esigenze espresse dal proponente vengano rispettate dai prodotti e processi da sviluppare.

### 3.3.2 Descrizione

Il *Piano di Qualifica* è il documento dedicato alla gestione della qualità. In esso sono descritti metriche e standard con le quali misurare e valutare la qualità di prodotti e processi.

### 3.3.3 Attività di processo

Si possono individuare tre attività principali nel processo di gestione di qualità:

- **Pianificazione:** definire obiettivi di qualità, le strategie per raggiungerli e le risorse necessarie;
- **Valutazione:** applicare quanto pianificato, misurando i risultati;
- **Reazione:** analizzando i risultati ottenuti con lo scopo di attuare miglioramenti o sanare situazioni non desiderate.

### 3.3.4 Controllo di qualità

Per essere sicuri di arrivare alla qualità desiderata, ogni membro deve essere in grado di:

- Comprendere gli obiettivi da raggiungere;
- Individuare eventuali errori;
- Stimare in termini di valore, dimensione e complessità le task;
- Produrre risultati concreti e quantificabili.

### 3.3.5 Denominazione metriche e obiettivi

Per la denominazione delle metriche è stato adottato il formato:

**M[Tipologia][Numero]**

mentre per la denominazione degli obiettivi di qualità è stato adottato il formato:

**Q[Tipologia][Numero]**

dove:

- **[Tipologia]:** indica la tipologia a cui si riferisce la metrica, può assumere tre valori:
  - **PC:** relativa ai processi;
  - **QP:** relativa ai prodotti;
  - **TS:** relativa ai test.
- **[Numero]:** indica il numero progressivo della metrica, parte da 1.

## 3.4 Verifica

### 3.4.1 Scopo

Lo scopo è definire come attuare il processo di verifica, per accertarsi che non ci siano errori durante lo sviluppo del prodotto, che la redazione della documentazione e che i requisiti in oggetto vengano rispettati.

### 3.4.2 Descrizione

La verifica viene applicata ad ogni processo in esecuzione. Per il processo di verifica ci si affida all'analisi e ai test. L'analisi si divide in due tipologie:

- **Statica:** non richiede l'esecuzione dell'oggetto in verifica, perciò è applicabile ad ogni prodotto. Accerta il rispetto della normativa e l'assenza di errori studiando il codice sorgente o la documentazione;
- **Dinamica:** richiede l'esecuzione dell'oggetto in verifica, applicabile perciò solo al codice. Fa largo uso di test.

### 3.4.3 Verifica della documentazione

Si utilizza un'analisi statica. Si possono utilizzare strumenti automatici oppure si può fare a mano (desk check) attraverso due metodi:

- **Walkthrough:** che consiste in un controllo completo del documento tramite una lettura ad ampio spettro;
- **Inspection:** che consiste in un controllo specifico tramite una lettura mirata del documento.

#### 3.4.3.1 Walkthrough

Attività svolta dal Verificatore, viene di norma adottata nelle fasi iniziali. Risulta essere molto onerosa e quindi va ridotto l'uso il prima possibile. Nel caso del nostro gruppo, questo metodo verrà adottato fino a che non sarà disponibile una lista di controllo.

#### 3.4.3.2 Inspection

Attività svolta dal Verificatore, sfrutta una lettura mirata per trovare gli errori tramite una lista di controllo, compilata seguendo gli errori più comuni nelle varie verifiche in modalità Walkthrough. Rispetto la tecnica Walkthrough, questo metodo è meno oneroso e da preferire, verrà utilizzata dal nostro gruppo quando sarà disponibile una lista di controllo esaustiva.

### 3.4.4 Verifica del codice

Si utilizza sia l'analisi statica che quella dinamica. In particolare con:

- L'analisi statica si controlla la bontà del codice, sia dal punto di vista della correttezza che del rispetto delle norme di buona programmazione definite dal gruppo (vedi TODO);
- L'analisi dinamica si controlla la presenza o meno di bug durante l'esecuzione del software prodotto.

#### 3.4.4.1 Test

I test sono l'attività fondamentale dell'analisi dinamica. Servono per dimostrare che il programma funzioni e svolga ciò per cui è stato sviluppato. I test si dividono in quattro categorie, in base all'oggetto in verifica e allo scopo:

- **Test d'unità:** verificano una singola unità di codice, la più piccola parte che ha senso verificare (es. singola procedura);
- **Test d'integrazione:** verificano la correttezza delle interfacce. Si vuole stabilire il corretto funzionamento delle varie componenti, una volta passato il test d'unità, aggregandole man mano e verificando il funzionamento nel complesso;
- **Test di sistema:** verifica l'applicazione nella sua interezza. Venendo dopo il test d'integrazione, lo scopo è verificare che le componenti non solo sono compatibili, ma che lo scambio di dati tra interfacce e le varie interazioni siano conformi. Inoltre, così si controlla che i requisiti siano stati soddisfatti;
- **Test di regressione:** verifica l'applicazione dopo aver apportato modifiche al sistema. Lo scopo è controllare nuove funzionalità non testate e, al tempo stesso, garantire che il codice già presente e testato in precedenza non subisca alterazioni al suo comportamento.

#### 3.4.4.2 Nominazione dei test

Ogni test verrà identificato con un codice:

**T[Tipologia][Id]**

In particolare:

- **Tipologia:** Indica la tipologia del test:
  - U: Unità;

- **I:** Integrazione;
- **S:** Sistema;
- **R:** Regressione.

- **Id:** codice numerico per identificare i test dello stesso tipo, parte da 1.

### 3.4.5 Verifica dei requisiti

Vengono applicate Walkthrough e Inspection per controllare validità e coerenza con quanto dichiarato e descritto all'interno dell'*Analisi dei Requisiti*, oltre a quanto dichiarato al proponente.

### 3.4.6 Metriche

Per la valutazione del processo di verifica si useranno le seguenti metriche:

- **MQP04 Code coverage;**
- **MQP10 Branch coverage;**
- **MQP11 Successo dei test.**

Per maggiori informazioni consultare §C.

## 3.5 Validazione

### 3.5.1 Scopo

Lo scopo è stabilire se il prodotto è in grado di soddisfare l'obiettivo per il quale è stato creato. Una validazione con esito positivo certifica che il software soddisfa e sia conforme ai requisiti del proponente.

### 3.5.2 Descrizione

Questo processo avviene dopo il processo di verifica, validando i risultati dei test e garantendo che siano conformi ai requisiti del proponente. Il compito spetta al Responsabile che controlla i risultati ottenuti e può:

- Accettare e approvare il prodotto;
- Rifiutare e chiedere un'ulteriore verifica con delle nuove indicazioni.

## 4 Processi Organizzativi

### 4.1 Gestione di processo

#### 4.1.1 Scopo

Secondo lo standard ISO-12207:1995 la gestione di processo contiene le attività e i compiti generici utili per la gestione dei rispettivi processi. Vengono individuate le seguenti attività:

1. Inizializzazione e definizione dello scopo;
2. Pianificazione e stima dei tempi, delle risorse, dei costi, assegnazione di compiti e responsabilità;
3. Esecuzione e controllo;
4. Revisione e valutazione;
5. Determinazione della fine del processo.

#### 4.1.2 Obiettivi

- Semplificare e gestire la comunicazione tra i membri del gruppo e l'esterno;
- Coordinare l'assegnazione dei ruoli e compiti;
- Monitorare il lavoro del gruppo e pianificare le attività da svolgere;
- Definire le linee guida generali per la formazione dei membri.

#### 4.1.3 Coordinamento

L'attività di coordinamento è responsabile della gestione delle comunicazioni interne, esterne e delle riunioni.

##### 4.1.3.1 Comunicazione

Le comunicazioni avvengono su due piani diversi: tra i membri del gruppo (interno), tra i membri del gruppo e uno o più soggetti esterni (esterno). I soggetti esterni si identificano in:

- **Proponente:** l'azienda *Zero12*;
- **Committenti:** Prof. *Tullio Vardanega* e Prof. *Riccardo Cardin*.

**Comunicazione interna** I principali mezzi di comunicazione tra membri del gruppo sono: Slack e Telegram.

In particolare, per Slack sono stati creati dei canali appositi per argomenti ed i membri del gruppo dovranno intrattenere le discussioni nei canali ritenuti più appropriati, per evitare confusione e/o ammassi incoerenti di messaggi. La creazione dei canali è stata fatta dal Responsabile di Progetto ed essi saranno destinati a mutare nel tempo per accomodare le esigenze del gruppo. Telegram viene usato per la componente più informale delle discussioni, per le decisioni meno importanti o in caso di problemi con il funzionamento di Slack (da considerarsi comunque una possibilità remota).

Per quanto riguarda la comunicazione interna tramite videochiamata (tipicamente le riunioni) lo strumento principale è Discord, scelto poiché semplice, conosciuto da tutti i membri del gruppo e multi-piattaforma. In alternativa verrà utilizzato Google Meet o Zoom, entrambi strumenti con i quali tutti i membri del gruppo hanno già avuto esperienza.

**Comunicazione esterna** Per la comunicazione con i soggetti esterni verrà utilizzato un indirizzo e-mail apposito: *dreamteam.unipd@gmail.com*

. Tutti i membri del gruppo avranno accesso alla casella di posta elettronica e saranno tenuti a verificare e notificare il gruppo circa la ricezione di nuovi messaggi, mentre la stesura e l'invio di un nuovo messaggio spetterà al Responsabile di Progetto. Prima dell'invio del messaggio, quest'ultimo sarà sottoposto ad una breve verifica e approvazione da parte del gruppo. Si sono decise le seguenti convenzioni per quanto riguarda la struttura delle e-mail:



- L'oggetto dovrà terminare con la dicitura "| SWE - Unipd";
- In generale il corpo dovrà mantenere un tono il più possibile formale ed il contenuto dovrà essere chiaro e conciso;
- Il corpo dovrà terminare con la firma "*DreamTeam*".

#### 4.1.3.2 Riunioni

Le riunioni potranno essere interne od esterne. Prima di ogni riunione verrà nominato un segretario che ha lo scopo di far rispettare l'ordine del giorno e dirigere la discussione, tenendo traccia dei punti salienti per poi poter redigere il verbale.

**Riunioni interne** Alle riunioni interne parteciperanno solo i membri del gruppo, per essere considerate valide dovranno essere presenti almeno quattro dei membri del gruppo. Prima di ogni riunione il Responsabile di progetto dovrà:

- fissare la data e l'orario;
- definire l'ordine del giorno;
- nominare il segretario;
- comunicare tutto quanto detto sopra (o eventuali variazioni) ai membri con ragionevole anticipo.

Gli incontri saranno svolti a cadenza settimanale e, in caso di necessità, anche più frequentemente. Ogni membro può rivolgersi al Responsabile di progetto per richiedere una riunione interna, proponendo l'ordine del giorno, poi se ritenuta necessaria il Responsabile si attiverà per l'organizzazione dell'incontro.

In ogni caso i membri del gruppo sono liberi di indire riunioni informali tra gli interessati, le persone coinvolte gestiranno orari, date e temi di discussione tra di loro come meglio credono senza intervento del Responsabile di progetto. Generalmente la riunione non sarà considerata valida e non sarà prodotto un verbale.

**Riunioni esterne** Alle riunioni esterne parteciperanno sia i membri del gruppo che soggetti esterni (proponente e/o committente). La richiesta di una riunione potrebbe provenire da entrambe le parti (soggetti esterni o Responsabile di progetto) e la piattaforma standard utilizzata sarà GMeet/Zoom, anche se il soggetto esterno è libero di scegliere la piattaforma che più desidera (non sono escluse le riunioni in presenza). In ogni caso il Responsabile di progetto dovrà nominare un segretario, incaricato della stesura del verbale.

#### 4.1.4 Pianificazione

Secondo lo standard ISO-12207:1995 l'attività di pianificazione prevede la preparazione delle risorse necessarie per l'avvio, l'esecuzione e la gestione di un processo. Spetterà quindi al Responsabile di progetto individuare i materiali, il tempo, il personale e le tecnologie necessarie, verificandone la disponibilità e l'adeguatezza. Parte fondamentale è l'identificazione dei ruoli di progetto e l'assegnazione dei compiti.

##### 4.1.4.1 Ruoli di progetto

Vengono identificati 6 ruoli di progetto che i membri dovranno assumere:

- Responsabile di Progetto;
- Amministratore di Progetto;
- Analista;
- Progettista;
- Programmatore;
- Verificatore.

Importante notare che ogni membro ricoprirà almeno una volta ogni singolo ruolo e andranno evitati i conflitti di interesse tra ruoli (es. una persona non potrà essere sia redattore che verificatore delle stesse parti)

**Responsabile di progetto** Il Responsabile di progetto è una figura fondamentale, incaricato di rappresentare il team all'esterno (proponente e committente), di guidare e coordinare il team al raggiungimento degli obiettivi di progetto nel rispetto dei costi e tempi concordati. Sarà un ruolo presente per tutta la durata del progetto. In particolare il suo ruolo prevede:

- responsabilità rispetto alle decisioni prese e dei documenti approvati;
- coordinamento dei membri del team e dei compiti da svolgere;
- mantenimento delle relazioni del team con i soggetti esterni;
- valutazione dei rischi e stima dei costi;
- responsabilità sulla pianificazione nel rispetto delle scadenze e dell'allocazione delle risorse.

**Amministratore di progetto** L'Amministratore di Progetto gestisce e controlla l'ambiente di lavoro, in particolare quelli che sono gli strumenti utilizzati e le regole che il team è tenuto a rispettare per tutta la durata del progetto. Il suo obiettivo è di favorire la produttività dei membri del gruppo. Generalmente presente per tutta la durata del progetto. I compiti specifici di questa figura sono:

- definire le norme e le procedure alla base del lavoro;
- regolare le infrastrutture e i servizi utili per lo svolgimento dei processi ;
- gestire il versionamento dei prodotti e la loro configurazione;
- individuare strumenti utili a migliorare e/o automatizzare i processi;
- gestire la documentazione di progetto.

**Analista** L'Analista è un ruolo fondamentale che partecipa al progetto soprattutto nella fase iniziale, con lo scopo di comprendere appieno e semplificare il problema, riducendolo ai suoi concetti chiave. I suoi compiti sono:

- studiare e definire il problema;
- identificare quali sono i vari bisogni e richieste, dalla quale saranno estratti dei requisiti;
- redigere l'analisi dei requisiti.

**Progettista** Il Progettista ha il compito di produrre una soluzione che soddisfi il problema e i bisogni individuati dagli analisti nell'Analisi dei Requisiti. I suoi compiti sono:

- sviluppare una soluzione che rispetti tutti e soli i requisiti individuati;
- produrre un'architettura adatta, in coerenza con i bisogni da soddisfare, che sia affidabile e consistente;
- perseguire il più possibile efficienza e riusabilità;
- impegnarsi a rimanere nei costi prestabiliti.

**Programmatore** Il Programmatore è responsabile della codifica e si occupa di implementare l'architettura sviluppata dal Progettista. In particolare deve:

- produrre codice che sia il più possibile orientato alla futura manutenzione, versionabile e documentato;
- scrivere il manuale utente per il codice prodotto.

**Verificatore** Il Verificatore ha il ruolo di controllare ciò che viene prodotto dagli altri membri del team. In particolare deve:

- controllare e individuare eventuali errori del prodotto in esame (fase di revisione);
- segnalare eventuali errori all'autore (o al responsabile) del prodotto analizzato.

#### 4.1.4.2 Gestione dei ticket

La gestione dei ticket è un'attività fondamentale in quanto serve al Responsabile di Progetto per assegnare i vari compiti ai membri del team, mentre permette a quest'ultimi di gestire meglio il carico di lavoro mostrando anche la progressione del progetto.

Il servizio di ticketing scelto è quello offerto da GitHub, tramite le issue. Il motivo principale di questa decisione è la comodità della piattaforma, evitando l'introduzione di strumenti esterni aggiuntivi.

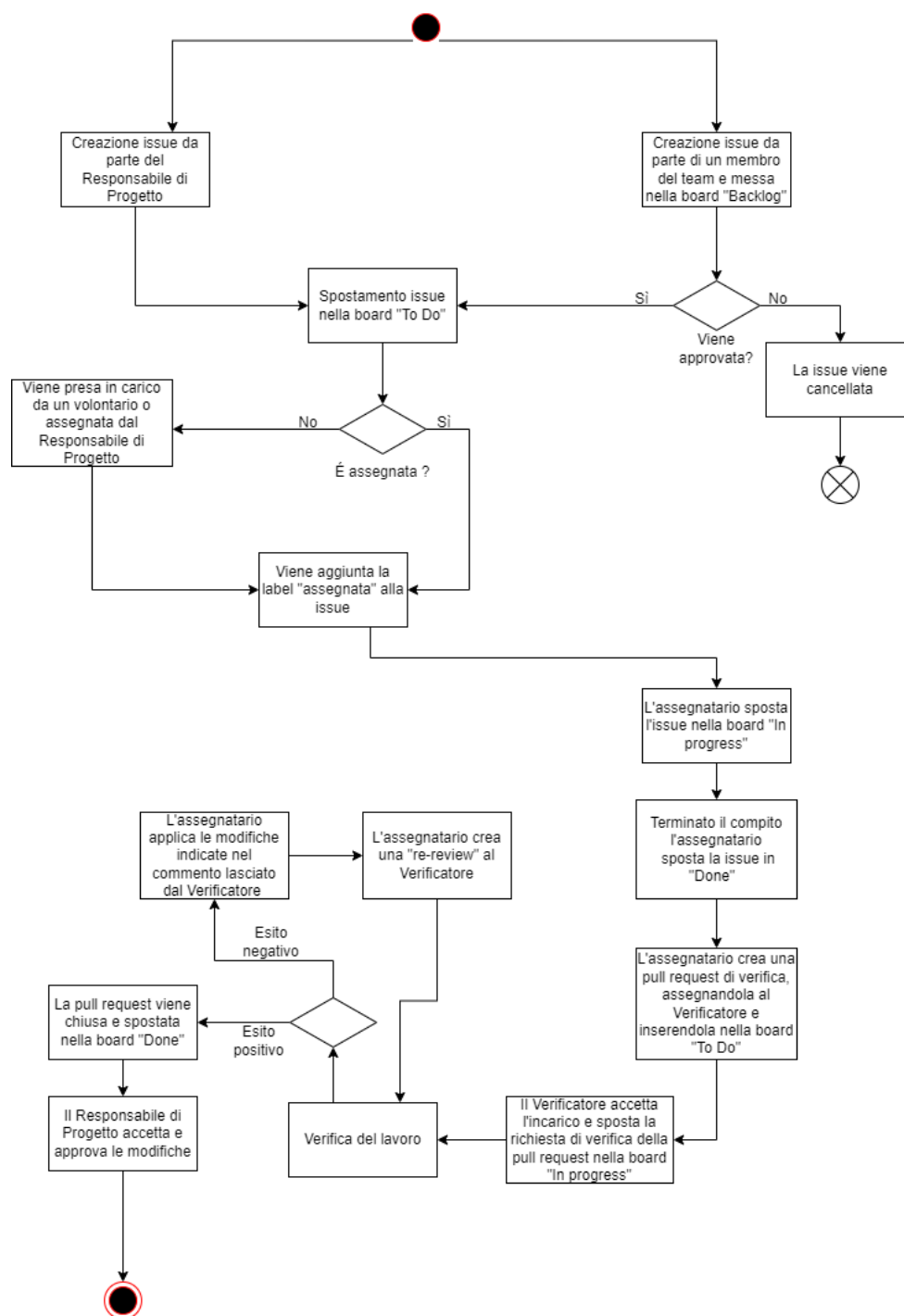


Figura 2: Attività di gestione dei ticket

Note:

- Viene data a tutti i membri la possibilità di proporre un ticket per snellire la procedura;

- Il Responsabile di Progetto può decidere di scomporre un ticket in "sotto-ticket" quando ritiene che questo sia troppo complesso;
- In generale i membri sono tenuti a dare titoli e descrizioni concise e coerenti dei ticket che creano, non saranno accettati ticket con corpo vuoto o ticket senza scadenza;

#### 4.1.4.3 Metriche

Le metriche adottate per la valutazione del processo di pianificazione sono le seguenti:

- **MPC02 Budgeted cost of work scheduled;**
- **MPC03 Actual cost of work performed;**
- **MPC05 Schedule variance;**
- **MPC06 Budget variance;**

Per maggiori informazioni consultare l'appendice B.

## 4.2 Formazione dei membri del team

Il processo di formazione ha lo scopo di assicurare che ogni membro abbia le conoscenze necessarie per svolgere i compiti che gli vengono assegnati e deve garantire il mantenimento di un personale competente nel tempo.

### 4.2.1 Obiettivi

Il processo mira al mantenimento costante (auspicabilmente per tutta la durata del progetto) di membri competenti ed esperti e garantisce quindi qualità di lavoro in linea con le aspettative.

### 4.2.2 Formazione interna

Ogni membro dovrà provvedere alla propria formazione in maniera autonoma, approfondendo le proprie mancanze con lo studio personale. I membri più esperti potranno condividere le loro conoscenze e/o materiale con il resto del team. In ogni momento un membro che trova difficoltà nell'esecuzione di un compito può rivolgersi al Responsabile di Progetto che dovrà organizzare le attività necessarie per l'apprendimento. Si farà riferimento alla documentazione seguente, ma essendo non esaustiva è consigliato un approfondimento con materiale reperito di proprio conto:

- **L<sup>A</sup>T<sub>E</sub>X**: <https://www.latex-project.org/help/documentation/>;
- **GitHub**: <https://docs.github.com/>;
- **Git**: <https://git-scm.com/doc>;
- **Node.js** <sup>G</sup>: <https://nodejs.org/api/>;
- **AWS Cognito**: <https://docs.aws.amazon.com/cognito/latest/developerguide/>;
- **AWS Comprehend** <sup>G</sup>: <https://docs.aws.amazon.com/comprehend/>;
- **API Gateway**: <https://docs.aws.amazon.com/apigateway/latest/developerguide/>;
- **AWS Lambda**: <https://docs.aws.amazon.com/lambda/>.

### 4.2.3 Formazione esterna

L'azienda proponente zero12 ha deciso di offrire ai membri del team una formazione specifica sulle tecnologie da loro richieste. L'azienda fornirà di volta in volta risorse informative sul canale Slack apposito, con lo scopo di fornire già una minima preparazione prima degli incontri formativi. Ogni membro è tenuto ad uno studio personale del materiale condiviso prima di ogni incontro.



#### 4.2.4 Strumenti a supporto della gestione organizzativa

Il team userà i seguenti strumenti per supportare il processo di gestione organizzativa:

- **Slack:** strumento standard usato dal team per la comunicazione interna e con il proponente;
- **Telegram:** strumento di messaggistica usato dal team per decisione e questioni meno importanti;
- **Git:** strumento utilizzato per il versionamento;
- **GitHub:** strumento utilizzato per il versionamento e per il salvataggio di tutti i file prodotti dai membri del team;
- **GitHub Issues:** sistema integrato in GitHub usato per la gestione dei ticket;
- **Google Drive:** utilizzato per la condivisione di documenti che richiedono molti cambiamenti dove la modifica in tempo reale è richiesta;
- **Gmail:** servizio di posta elettronica scelto dal gruppo;
- **Zoom:** strumento usato all'inizio per fare riunioni tra membri del team;
- **Discord:** strumento standard usato per le riunioni tra i membri del team.

## A Standard ISO/IEC 15504 - SPICE

### A.1 Introduzione

Lo standard SPICE (*Software Process Improvement and Capability Determination*) è un insieme di standard tecnici per i processi di sviluppo software.

#### A.1.1 Classificazione processi

Nello standard SPICE i processi sono suddivisi in 5 categorie:

- Cliente-fornitore;
- Ingegneristico;
- Supporto;
- Gestionale;
- Organizzativo.

### A.2 Livelli di capability e attributi di processo

La *capability* viene definita come la capacità di un processo di raggiungere il suo scopo. Per ogni processo, SPICE definisce un livello di capability dai seguenti:

- **Livello 0 - Incomplete process:** processo non implementato, incapace di raggiungere i suoi obiettivi;
- **Livello 1 - Performed process:** processo implementato e in grado di raggiungere i suoi obiettivi, ma non è sottoposto nessun tipo di controllo;
- **Livello 2 - Managed process:** processo pianificato e sottoposto a controllo e correzione, gli obiettivi vengono raggiunti e sono tracciabili e verificati;
- **Livello 3 - Established process:** processo definito da standard e quindi regolamentato;
- **Livello 4 - Predictable process:** processo istanziato entro limiti ben definiti, viene monitorato in modo dettagliato con lo scopo di renderlo prevedibile e ripetibile;
- **Livello 5 - Optimizing process:** processo completamente definito e tracciato, soggetto ad analisi e miglioramento continui.

La capability di un processo è misurata tramite gli attributi di processo, lo standard definisce 9 attributi (dove il codice rappresenta il livello alla quale vengono applicati):

- **1.1 Process performance:** numero di obiettivi raggiunti;
- **2.1 Performance management:** livello di organizzazione degli obiettivi fissati;
- **2.2 Work product management:** livello di organizzazione dei prodotti rilasciati;
- **3.1 Process definition:** livello di adesione agli standard prefissati;
- **3.2 Process deployment:** livello di ripetibilità del processo;
- **4.1 Process measurement:** livello di efficacia di applicazione delle metriche al processo;
- **4.2 Process control:** livello di predicibilità delle valutazioni;
- **5.1 Process innovation:** misura gli aspetti positivi generati dei cambiamenti attuati dopo una fase di analisi;
- **5.2 Process optimization:** misura l'efficienza del processo, il rapporto tra i risultati ottenuti e le risorse impegnate.



Ogni processo è valutato tramite la seguente scala di valori che esprimono numericamente il grado di soddisfacimento dell'attributo:

- Not achieved (0 – 15%);
- Partially achieved (> 15 – 50%);
- Largely achieved (> 50 – 85%);
- Fully achieved (> 85 – 100%).

## B Metriche per la qualità del processo

### B.1 MPC01 SPICE

Metrica utilizzata per misurare la qualità dei processi impiegati e fornire una valutazione relativa ad essi. Lo standard è illustrato nel dettaglio all'interno dell'appendice A.

### B.2 MPC02 Budgeted cost of work scheduled

Indica una previsione della somma di budget spesa dal gruppo per il lavoro svolto dall'inizio del progetto fino alla data corrente. Questo valore è reperibile all'interno del Piano di Progetto.

### B.3 MPC03 Actual cost of work performed

Valore che indica la somma di tutte le spese realmente sostenute dal gruppo, questo valore deve essere minore o uguale a quanto preventivato nel Piano di Progetto

### B.4 MPC04 Budgeted cost of work performed

Indica il valore effettivo del prodotto ottenuto fino al momento in cui l'indice viene calcolato.

### B.5 MPC05 Schedule variance

Indica il discostamento percentuale tra la programmazione preventivata e quella effettiva. La formula è:

$$SV = \frac{100 * (BCWP - BCWS)}{BCWS};$$

- SV = Schedule variance;
- BCWP = Budgeted cost of work performed;
- BCWS = Budgeted cost of work scheduled;

### B.6 MPC06 Budget variance

Indice percentuale che mostra se le spese sostenute a partire dall'inizio del progetto fino al momento in cui viene calcolato rientrano nel budget di spesa previsto per la data corrente. La formula è:

$$BV = \frac{100 * (BCWS - ACWS)}{BCWS};$$

- BV = Budget variance;
- BCWS = Budgeted cost of work scheduled;
- ACWP = Actual cost of work performed.

### B.7 MPC07: Requirements stability

Indice percentuale che mostra come i requisiti cambiano nel tempo. La formula è:

$$RS = 1 - \frac{RM}{RT} * 100;$$

- RS = Requirements stability;
- RM = somma tra i requisiti modificati, eliminati e aggiunti;
- RT = totale dei requisiti iniziali.



## C Metriche per la qualità del processo

### C.1 MQP01 Indice di Gulpease

È un indice di leggibilità di un determinato testo. Calcola la lunghezza delle parole e delle frasi rispetto al numero totale delle lettere, per fare ciò considera: la lunghezza della parola e la lunghezza della frase rispetto al numero delle lettere. Il valore è un numero intero compreso tra 0 e 100. La formula è:

$$IDG = 89 + \frac{300 * NDF - 10 * NDL}{NDP};$$

- NDF = Numero dei frasi;
- NDL = Numero di lettere;
- NDP = Numero di parole.

### C.2 MQP02 Profondità di una gerarchia

È un valore intero che definisce quanto può essere profonda la gerarchia di una classe. Nel caso una gerarchia abbia una sola classe, allora il suo valore sarà pari a 1.

### C.3 MQP03 Numero parametri per metodo

Indica il numero intero di parametri che può avere un metodo. Un numero di parametri troppo elevato, indica che è necessario ridurre delle funzionalità associate al metodo a cui si fa riferimento. Inoltre, se si ha un numero elevato di parametri, la probabilità di avere più errori progettuali aumenta.

### C.4 MQP04 Code coverage

È la misura in percentuale di linee di codice eseguite in un test rispetto al totale delle linee di codice del sorgente di un programma. Un programma che ha un'elevata copertura del codice sorgente eseguito durante i test ha una minore possibilità di contenere bug.

### C.5 MQP05 Percentuale requisiti obbligatori soddisfatti

Indica la quantità percentuale dei requisiti obbligatori soddisfatti in rapporto al totale dei requisiti obbligatori. La formula è:

$$RS = \frac{ROS}{TRO} * 100;$$

- RS = Percentuale requisiti obbligatori soddisfatti
- ROS = Requisiti obbligatori soddisfatti;
- TRO = Totale requisiti obbligatori.

### C.6 MQP06 Complessità ciclomatica

La complessità ciclomatica è una metrica sviluppata da Thomas J. McCabe fornisce una misura quantitativa della complessità di un programma, essa è identificata dal numero di cammini linearmente indipendenti presenti nel grafo del flusso di controllo del programma. Un valore elevato di tale misurazione indica che il software possiede un comportamento poco predicibile, fattore che causa grandi rischi. La formula è:

$$CC = E - N + P;$$

- CC = Complessità ciclomatica;
- E = Numero di congiunzioni tra statement *gli archi di un grafo*;
- N = Numero di statement *nodipresentinelgrafo*;
- P = Numero delle componenti connesse da ogni nodo *per esecuzione sequenziale p = 2essendovi 1 predecessore e 1 successore*

### C.7 MQP07 Numero di bug

Numero di righe di codice del programma che potrebbero comportare un risultato diverso da quello previsto

### C.8 MQP08 Numero di code smell

Indica una serie di caratteristiche che il codice sorgente può avere e che sono generalmente riconosciute come probabili indicazioni di un difetto di programmazione

### C.9 MQP09 Linee di Commento per Linee di Codice

La formula è:

$$LCC = \frac{LDC}{LCD} * 100;$$

- LCC = Percentuale rapporto tra linee di commento e linee di codice di istruzioni;
- LDC = Linee di commento;
- LCD = Linee di codice.

### C.10 MQP10 Branch coverage

Percentuale di copertura di rami condizionali durante i test La formula è:

$$BC = \frac{RCC}{RCT} * 100;$$

- BC = Percentuale rapporto tra linee di commento e linee di codice di istruzioni;
- RCC = Rami condizionali coperti da test;
- RCT = Rami condizionali totali.

### C.11 MQP11 Successo dei test

Percentuali di test superati rispetto ai test implementati. La formula è:

$$SDT = \frac{NTS}{NTT} * 100;$$

- SDT = Percentuale di successo dei test;
- NTS = Numero di test superati;
- NTT = Numero di test totali.

### C.12 MQP12 Numero di bug

Indica il numero di vulnerabilità presenti nel codice sorgente del prodotto.