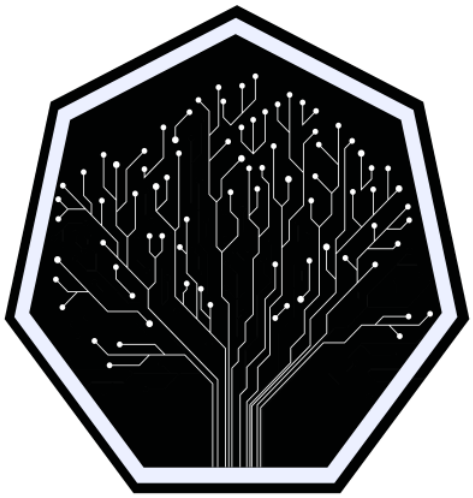

VRAM SOFTWARE

Progetto "Predire in Grafana"

Norme di Progetto

11 gennaio 2020

Versione	1.1.1
Approvazione	Corrizzato Vittorio
Redazione	Rampazzo Marco
	Corrizzato Vittorio
	Santagiuliana Vittorio
Verifica	Schiavon Rebecca
	Corrizzato Vittorio
	Spreafico Alessandro
Stato	Approvato
Uso	Interno
Destinato a	Zucchetti
	Prof. Tullio Vardanega
	Prof. Riccardo Cardin
Email di riferimento	vram.software@gmail.com

Descrizione

Questo documente contiene gli strumenti e le norme adottate dal gruppo *VRAM Software*

Registro delle modifiche

Versione	Data	Nominativo	Ruolo	Descrizione
1.1.1	2020-01-11	Corrizzato Vittorio	*Responsabile*	Approvazione documento.
0.8.4	2020-01-11	Santagiuliana Vittorio	*Amministratore*	Aggiornamento e revisione finale.
0.8.3	2020-01-11	Schiavon Rebecca	*Verificatore*	Verifica del documento finale.
0.8.3	2020-01-06	Schiavon Rebecca	*Verificatore*	Verifica delle sezioni scritte al giorno 2020-01-05.
0.8.3	2020-01-05	Santagiuliana Vittorio	*Amministratore*	Aggiornamento e revisione delle sezioni riguardanti *Piano di Progetto*, *Piano di Qualifica* e strumenti da utilizzare.
0.7.3	2020-01-02	Rampazzo Marco	*Amministratore*	Revisione delle sezioni scritte come da istruzioni verificatore.
0.7.2	2020-01-02	Santagiuliana Vittorio	*Verificatore*	Verifica delle sezioni scritte al giorno 2020-01-02.
0.7.2	2020-01-02	Rampazzo Marco	*Amministratore*	Stesura e revisione delle sezioni riguardanti gestione della qualità e strumenti da utilizzare.

0.6.2	2019-12-30	Rampazzo Marco	*Amministratore*	Stesura e revisione delle sezioni riguardanti *Analisi dei Requisiti* e *Piano di Qualifica*.
0.5.2	2019-12-23	Rampazzo Marco	*Amministratore*	Revisione delle sezioni scritte come da istruzioni verificatore.
0.5.1	2019-12-22	Spreafico Alessandro	*Verificatore*	Verifica delle sezioni scritte al giorno 2019-12-21.
0.5.1	2019-12-21	Rampazzo Marco	*Amministratore*	Stesura e revisione delle sezioni riguardanti *Analisi dei Requisiti*, *Piano di Progetto* e strumenti da utilizzare.
0.4.1	2019-12-04	Corrizzato Vittorio	*Verificatore*	Verifica delle sezioni scritte al giorno 2019-12-03.
0.4.1	2019-12-03	Toffoletto Massimo	*Amministratore*	Aggiornamento e revisione delle sezioni riguardanti *Studio di Fattibilità* e strumenti da utilizzare.
0.3.1	2019-11-23	Corrizzato Vittorio	*Amministratore*	Revisione delle sezioni scritte come da istruzioni verificatore.
0.3.0	2019-11-22	Schiavon Rebecca	*Verificatore*	Verifica delle sezioni scritte al giorno 2019-11-22.

0.3.0	2019-11-21	Corrizzato Vittorio	*Amministratore*	Stesura delle sezioni riguardanti processo di verifica e processi organizzativi.
0.2.0	2019-11-20	Corrizzato Vittorio	*Amministratore*	Stesura delle sezioni riguardanti documentazione e gestione della configurazione.
0.1.0	2019-11-19	Corrizzato Vittorio	*Amministratore*	Stesura delle sezioni riguardanti *Studio di Fattibilità* e strumenti da utilizzare.

Indice

Elenco delle tabelle

1 Introduzione

1.1 Scopo del Documento

Il presente documento ha come scopo la definizione di regole e convenzioni che stanno alla base del progetto$_G$ e che tutti i membri del gruppo devono seguire. In questo modo si garantisce consistenza e omogeneità in tutto il materiale del progetto$_G$ stesso. Infatti ogni componente del gruppo è obbligato a visionare questo documento e a rispettarne rigorosamente le norme al fine di lavorare secondo una metodologia coesa e uniforme.

1.2 Scopo del Prodotto

Il capitolato$_G$ C4 ha lo scopo di implementare un plug-in di Grafana$_G$ scritto in linguaggio Javascript che eseguirà gli algoritmi di Support Vector Machine SVM$_G$ o Regressione Lineare RL$_G$, i quali leggendo da un file JSON la loro configurazione saranno in grado di generare previsioni che potranno essere aggiunte al flusso di monitoraggio. Il plug-in dovrà prevedere il superamento di determinati livelli di soglia per generare un allarme, oppure permettere agli operatori del servizio Cloud$_G$ di generare segnalazioni dei punti critici che l'erogazione mette in evidenza. Il plug-in utilizzerà un applicativo esterno per addestrare gli algoritmi di SVM$_G$ ed RL$_G$. Dunque il plug-in permetterà un'azione di monitoraggio grazie alla quale gli operatori potranno intervenire con ogni cognizione di causa sul sistema.

1.3 Ambiguità e Glossario

Questo documento verrà corredato da un *Glossario v. 1.1.1* dove saranno illustrati i termini tecnici o altamente specifici per evitare ambiguità in essi. Le voci interessate saranno identificate da una 'G' a pedice.

1.4 Riferimenti

1.4.1 Riferimenti Informativi

1. **Standard ISO/IEC 12207:1995**: `https://www.math.unipd.it/~tullio/IS-1/2009/Approfondimenti/ISO_12207-1995.pdf`;
2. **Capitolato$_G$ d'appalto C4 - Predire in Grafana**: `https://www.math.unipd.it/~tullio/IS-1/2019/Progetto/C4.pdf`
3. **Ingegneria del software, Ian Sommerville, Pearson Education, Addison Wesley (Decima edizione)**;

4. **Guide to the Software Engineering Body of Knowledge(SWEBOK), 2004**: `http://www.math.unipd.it/~tullio/IS-1/2007/Approfondimenti/SWEBOK.pdf`;

5. **Slide L05 del corso Ingegneria del Software - Ciclo di vita del software**: `https://www.math.unipd.it/~tullio/IS-1/2016/Dispense/L05.pdf`;

2 Processi Primari

2.1 Fornitura

2.1.1 Scopo

Lo scopo del processo$_G$ di fornitura è stabilire norme, tempi e risorse necessarie a svolgere l'intero progetto$_G$. A seguito dell'analisi sui capitolati$_G$, verrà redatto il documento *Studio di Fattibilità* nel quale verrà manifestata la decisione del capitolato$_G$ scelto; sarà quindi necessario formalizzare un contratto, insieme all'azienda proponente, per la consegna del prodotto. Per fare quanto sopra descritto si devono svolgere le seguenti attività:

- Avvio;
- Contrattazione con il proponente;
- Pianificazione del lavoro;
- Esecuzione di quanto pianificato con controllo della qualità;
- Revisione e valutazione del prodotto;
- Completamento e consegna al proponente.

2.1.2 Aspettative

Il gruppo si aspetta di mantenere un dialogo costante con l'azienda proponente instaurando un rapporto collaborativo al fine di comprendere meglio le esigenze. In particolare ci si focalizza sui seguenti termini:

- Definizione degli elementi fondamentali su cui focalizzarsi per soddisfare le necessità del proponente;
- Definizione di vincoli e requisiti sui processi;
- Stima delle tempistiche e della pianificazione del lavoro;
- Verifica continua e chiarimento di eventuali dubbi;
- Accordo sulla qualifica del prodotto.

2.1.3 Descrizione

La sezione dei processi$_G$ di fornitura specifica le regole che devono essere rispettate affinché il nostro gruppo possa diventare fornitore del proponente Zucchetti.

2.1.4 Attività

2.1.4.1 Studio di fattibilità

Lo studio di fattibilità è un'attività che ha lo scopo di stilare le motivazioni per cui il gruppo ha preferito un capitolato$_G$ rispetto agli altri. Risulta quindi necessario individuare costi e benefici di ogni capitolato$_G$ al fine di determinare quello che meglio si bilancia con le aspettative del gruppo e le risorse a disposizione. Dunque gli analisti redigono il documento *Studio di Fattibilità* dove per ogni capitolato$_G$ vengono indicati:

- **Informazioni Generali**: elenco che presenta le seguenti informazioni:
  - Nome del progetto$_G$;
  - Proponente;
  - Committente.
- **Descrizione**: breve esposizione dell'argomento del capitolato$_G$;
- **Obiettivi di Progetto**: descrizione dettagliata centrata sugli obiettivi da raggiungere;
- **Vincoli**: elenco dei vincoli imposti dal proponente per la realizzazione del progetto$_G$;
- **Tecnologie Interessate**: elenco e descrizione sintetica delle tecnologie che dovranno essere impiegate nello svolgimento;
- **Aspetti positivi**: elenco delle principali motivazioni che porterebbero il gruppo a scegliere il capitolato$_G$;
- **Criticità e fattori di rischio**: elenco delle principali motivazioni che porterebbero il gruppo a non scegliere il capitolato$_G$;
- **Conclusioni**: sintesi delle motivazioni per cui il capitolato$_G$ è stato scelto o è stato escluso.

2.1.4.2 Piano di progetto

Il *Piano di Progetto* è un documento redatto dal responsabile in collaborazione con gli amministratori ed ha lo scopo di pianificare le attività per la realizzazione del progetto e individuare le risorse disponibili da assegnare a ciascuna di esse. Questo documento è composto da:

- **Introduzione**: descrizione generale del documento che specifica anche lo scopo del documento e del prodotto e il calendario delle attività;

- **Analisi dei rischi**: analisi dettagliata dei rischi che potrebbero sorgere durante il ciclo di sviluppo del progetto$_G$. In allegato vengono forniti stime probabilistiche, livelli di rischio e modalità previste per fare prevenzione o per affrontarli qualora non sia stato possibile evitarli;
- **Modello di sviluppo**: descrizione del modello di sviluppo selezionato per lo svolgimento del progetto$_G$;
- **Pianificazione**: specifica dei tempi e dei ruoli per ogni attività individuata al fine di affrontare al meglio le scadenze del capitolato$_G$. Essa viene limitata dalla quantità di risorse;
- **Preventivo e Consultivo**: stima dei costi necessari per lo svolgimento di ogni scadenza prevista e conseguente costruzione del preventivo per il progetto$_G$. Stesura di un *Verbale* interno su cui riportare l'andamento effettivo rispetto a quanto previsto;
- **Organigramma**: presentazione del nostro organigramma.

2.1.4.3 Piano di qualifica

Questa attività prevede la stesura del documento *Piano di Qualifica* che viene redatto dai verificatori. Esso ha lo scopo di descrivere strategie e tecniche di verifica e validazione$_G$ che devono essere applicate dai verificatori nelle loro attività. Questo documento è composto da:

- **Introduzione**: descrizione generale del documento che specifica anche lo scopo del documento e del prodotto;
- **Qualità di Processo**: individuazione dei processi da adottare in riferimento ad uno standard e delle metriche per la misurazione e il monitoraggio della loro qualità;
- **Qualità di Prodotto**: individuazione degli obiettivi posti sul prodotto che sono necessari per raggiungere una buona qualità e delle metriche per misurarla;
- **Descrizione dei test**: individuazione dei test a cui sottoporre il prodotto per garantire che i requisiti vengano soddisfatti;
- **Standard di Qualità**: elenco degli standard di qualità scelti;
- **Resoconto delle attività di verifica**: resoconti dei risultati dati dalle metriche di qualità calcolate per ogni attività;

- **Valutazioni per il miglioramento**: esposizione dei problemi rilevati e delle soluzioni da attuare per migliorare.

2.1.5 Strumenti

Durante il Processo$_G$ di Fornitura sono utilizzati i seguenti strumenti della suite di Microsoft.

2.1.5.1 Microsoft Excel

Per svolgere semplici calcoli, creazione di grafici, diagrammi e tabelle si è scelto di utilizzare il software Microsoft Excel.

2.1.5.2 Microsoft Project

Per realizzare il diagramma di Gantt necessario per la pianificazione, in particolare di tempi, risorse e analisi dei carichi di lavoro, si è scelto di utilizzare Microsoft Project.

2.2 Sviluppo

2.2.1 Scopo

Lo scopo del processo$_G$ di sviluppo è svolgere le attività necessarie per la realizzazione del prodotto richiesto.

2.2.2 Aspettative

Il gruppo si è posto le seguenti aspettative:

- Stabilire gli obiettivi di sviluppo;
- Stabilire i vincoli tecnologici e di progettazione$_G$;
- Realizzare un prodotto che soddisfa i requisiti imposti dai proponenti e superi i test per avere una buona qualità.

2.2.3 Descrizione

Il processo$_G$ di sviluppo è suddiviso nelle seguenti attività:

- **Analisi dei Requisiti**;
- **Progettazione**;
- **Codifica**.

2.2.4 Attività

2.2.4.1 Analisi dei Requisiti

Scopo

Lo scopo dell' *Analisi dei Requisiti* è individuare ed analizzare i requisiti del progetto$_G$ al fine di:

- Individuare lo scopo del lavoro;
- Definire un'analisi dettagliata e precisa del dominio dell'applicazione e dei requisiti che i progettisti dovranno seguire;
- Definire dei riferimenti per le attività di controllo dei test da fornire ai verificatori;
- Fornire una base solida da cui poter effettuare raffinamenti successivi e fornire un miglioramento continuo del progetto$_G$.

Aspettative

L'obiettivo è stilare una documentazione formale sui requisiti di progetto$_G$ richiesti dal proponente.

Descrizione

L'analisi dei requisiti viene redatta eseguendo le seguenti attività:

- **Comprensione del capitolato$_G$**: eseguita attraverso la lettura e l'analisi della documentazione fornita;
- **Comunicazioni con il proponente**: eseguita al fine di migliorare l'analisi del progetto$_G$;
- **Comunicazioni interne al gruppo**;
- **Analisi dei possibili casi d'uso$_G$**.

Casi d'uso$_G$

Un caso d'uso definisce un insieme di scenari con un obiettivo finale in comune per un attore. Sono ottenuti attraverso la valutazione di ogni requisito e descrivono l'insieme delle funzionalità fornite dal sistema dal punto di vista degli utenti. Per descrivere ogni caso d'uso viene utilizzata la seguente struttura:

- **Codice Identificativo**;
- **Titolo**;
- **Attori primari**;
- **Attori secondari**;
- **Descrizione**;
- **Precondizione**;
- **Post-condizione**;
- **Scenario principale**;
- **Scenario alternativo**;
- **Inclusioni** (ove presenti);
- **Estensioni** (ove presenti);
- **Generalizzazioni** (ove presenti);
- **Diagramma UML**.

Il codice identificativo sarà scritto in questo formato:
UC[codice_padre].[codice_figlio]
Dove:

- **codice_padre:** numero che identifica univocamente i casi d'uso$_G$;
- **codice_figlio:** numero progressivo che identifica i sotto-casi;

Requisiti
Per descrivere un requisito viene utilizzata la seguente struttura:

- Codice Identificativo;
- Classificazione;
- Descrizione;
- Fonti.

Il **Codice Identificativo** sarà scritto in questo formato:
R[Importanza][Tipologia][Codice]
Dove:

- **Importanza** può assumere i seguenti valori:
  - 1: requisito obbligatorio;
  - 2: requisito desiderabile;
  - 3: requisito opzionale.
- **Tipologia** può assumere i seguenti valori:
  - F: Funzionale;
  - Q: Prestazionale;
  - P: Qualitativo;
  - V: Vincolo.
- **Codice**: numero progressivo identificativo strutturato nel formato: [codice_padre].[codice_figlio]

Le **Fonti** possono essere:

- capitolato$_G$: il requisito è stato quindi individuato dalla lettura del capitolato$_G$;
- interno: il requisito è stato individuato ed aggiunto in seguito ad un'analisi interna;
- caso d'uso: il requisito è stato individuato dallo studio di un caso d'uso;
- proponente: il requisito è stato individuato in seguito ad un colloquio con il proponente.

I diagrammi UML$_G$ vengono realizzati usando la versione 2.0 del linguaggio.

2.2.4.2 Progettazione

Scopo

Lo scopo della progettazione$_G$ è descrivere una soluzione soddisfacente per gli stakeholder$_G$. Questa attività definisce l'architettura logica del prodotto software richiesto, a partire dall'*Analisi dei Requisiti* Viene quindi definita l'architettura del prodotto$_G$ finale che dovrà:

- Soddisfare i requisiti individuati nell'*Analisi dei Requisiti*;
- Essere comprensibile;
- Essere modulare;
- Definire le proprie parti con specifiche chiare e coese;
- Garantire robustezza al fine di riuscire a gestire errori improvvisi;
- Essere organizzata per facilitare cambiamenti futuri.

Aspettative

La progettazione$_G$ ha come risultato finale la realizzazione dell'architettura del sistema.

Descrizione

La progettazione$_G$ è composta da due sezioni:

- Technology baseline$_G$;
- Product baseline$_G$.

Tecnology Baseline

Nella Tecnology Baseline$_G$ si trovano:

- **Tecnologie Utilizzate**: descrizione con l'indicazione di vantaggi e svantaggi e l'utilizzo;
- **Design pattern**: descrizione del design pattern utilizzati all'interno dell'architettura. Per ciascuno di essi deve essere fornito un diagramma che ne definisce la struttura e una descrizione;
- **Diagrammi UML**:

  - **Diagrammi delle Classi**;
  - **Diagrammi dei Package**;
  - **Diagrammi di Attività**;
  - **Diagrammi di Sequenza**.

- **Tracciamento delle Componenti**: riferimento di ogni requisito al componente che lo soddisfa;
- **Test di Integrazione**: necessari per verificare che l'unione delle parti funzioni correttamente.

Product Baseline

Nella Product Baseline$_G$ sono presenti:

- **Descrizione delle Classi**: descrizione degli obiettivi e delle funzionalità di ogni classe;
- **Tracciamento delle Classi**: riferimento di ogni requisito alla classe che lo soddisfa;
- **Test di Unità**: necessari per verificare il corretto funzionamento di ogni singola parte del software.

2.2.4.3 Codifica

Scopo

Lo scopo di questa sezione è dare una norma per definire tutte le regole di codifica per garantire la leggibilità e la manutenibilità del codice.

Aspettative

L'obiettivo della codifica è creare un prodotto coerente con i requisiti e le aspettative del proponente, il cui codice sia leggibile, uniforme e permetta di eseguire più facilmente le attività di manutenzione e verifica. Dunque i programmatori sono tenuti a seguire tali regole durante l'attività di implementazione.

Descrizione

La codifica consiste nella scrittura del codice per realizzare il prodotto$_G$. La sua struttura dovrà rispettare le regole definite nel *Piano di Qualifica* al fine di garantire una buona qualità.

Stile di Codifica

In seguito vengono elencate le norme da rispettare nel processo$_G$ di codifica:

- **Lingua**: la lingua con cui vengono scritti codice e commenti deve essere l'inglese;
- **Indentazione**: i blocchi annidati devono essere indentati con una tabulazione di quattro spazi;
- **Nomenclatura**: i nomi di classi, metodi e variabili devono essere univoci ed esplicativi;
- **Classi**: il nome di ciascuna classe deve iniziare con la lettera maiuscola;
- **Costanti**: il nome delle costanti devono essere scritti usando solo lettere maiuscole;
- **Metodi**: il nome di ciascun metodo deve iniziare con una lettera minuscola. Nei metodi con nome composto, le iniziali di ogni singola parola che li compone deve essere una lettera maiuscola seguendo la pratica CamelCase;
- **Ricorsione**: l'uso della ricorsione è da evitare ove possibile;
- **Condizioni**: l'uso di condizioni multiple è da evitare ove possibile per limitare la complessità delle singole espressioni;
- **Parentesizzazione**: le parentesi di delimitazione dei costrutti devono essere inserite in linea;
- **Struttura dei metodi**: ogni metodo deve avere uno ed un solo compito e il suo contenuto deve essere il più possibile breve in termini di righe di codice.

2.2.5 Metriche di qualità

Scopo

Lo scopo di questa sezione è quello di definire delle metriche$_G$ per valutare in modo quantitativo la qualità delle attività che si svolgono durante il processo$_G$ di sviluppo.

Analisi dei Requisiti

I parametri utilizzati per valutare la qualità dell'analisi dei requisiti sono:
La percentuale di requisiti obbligatori soddisfatti (**PROS**), calcolata con la seguente formula: $\frac{requisiti\ obbligatori\ soddisfatti}{requisiti\ obbligatori\ totali}$;
La percentuale di requisiti desiderabili soddisfatti (**PRDS**), calcolata con la seguente formula: $\frac{requisiti\ desiderabili\ soddisfatti}{requisiti\ desiderabili\ totali}$.

2.2.6 Strumenti

Nella valutazione dei costi da affrontare per lo sviluppo del software e per la formazione del personale, abbiamo già identificato le principali tecnologie e strumenti che verranno utilizzare durante il progetto$_G$.

2.2.6.1 TeXstudio

Per la stesura dell'*Analisi dei Requisiti* viene utilizzato TeXstudio come ambiente di sviluppo.

2.2.6.2 SonarJS

Per l'analisi statica$_G$ del codice viene utilizzato il code analyzer SonarJS$_G$ su piattaforma di SonarCloud.

2.2.6.3 WebStorm

Per la codifica viene utilizzato WebStorm, un ambiente di sviluppo integrato per JavaScript che offre piena compatibilità con Windows, Linux e SonarJS$_G$.

2.2.6.4 LucidChart

Per la realizzazione dei diagrammi UML$_G$.

3 Processi di supporto

3.1 Documentazione

3.1.1 Descrizione

In questo capitolo verrà illustrato come il gruppo ha gestito il processo$_G$ di redazione della documentazione utile ai fini del progetto$_G$. Si andranno perciò a chiarire le fasi di costruzione del documento, la sua struttura, gli standard tipografici utilizzati e gli strumenti usati per la stesura.

3.1.2 Previsioni

Tale sezione si prefigge due obiettivi:

- standardizzare la scrittura della documentazione al fine di offrire degli elaborati consistenti e coerenti tra di loro;
- dare delle linee guida per la produzione di documenti corretti e validi.

3.1.3 Obiettivi

Questo documento stabilisce delle norme per la produzione e la verifica dei prossimi elaborati. Tali indicazioni saranno condivise e seguite da tutti gli elementi del gruppo.

3.1.4 Ciclo di vita dei documenti

Ogni documento, durante il suo ciclo di vita, attraversa tre fasi principali:

- **Stesura**: in questo stadio viene effettuata la redazione del documento seguendo le norme indicate e i template disponibili. Il responsabile, a seguito di un'analisi preliminare, divide (se necessario) il lavoro tra vari redattori che procederanno alla compilazione.
- **Verifica**: terminata la fase di stesura i redattori sottometteranno l'elaborato al controllo dei verificatori. I verificatori, nominati in precedenza dal responsabile, analizzeranno il documento. Nel caso in cui si evidenzino delle criticità segnaleranno le dovute modifiche ai redattori, in alternativa comunicheranno al responsabile la conformità del materiale.
- **Approvazione**: alla fine il documento viene vagliato dal responsabile che ne approva ufficialmente il rilascio.

3.1.5 Struttura dei documenti

3.1.5.1 Template LaTeX

Per la produzione di documenti consistenti e coerenti tra loro il gruppo ha deciso di creare un template LaTeXche permette, al momento della stesura, di concentrarsi unicamente sul contenuto e di non impegnare troppo tempo nella strutturazione della pagina. Il template è formato da più parti, tra cui:

- un file per la corretta formattazione della prima pagina;
- un file contenente i pacchetti LaTeXutilizzati per una migliore rappresentazione dei contenuti;
- un file per modificare i comandi personalizzati, i quali garantiscono una maggiore modularità del template;
- due file per la gestione delle tabelle.

3.1.5.2 Prima pagina

Il frontespizio è strutturato in questo modo:

- **Logo del gruppo**: corredato dal nome, il tutto centrato orizzontalmente;
- **Capitolato**$_G$: nome del progetto$_G$ scelto dal gruppo (*Predire in Grafana*), centrato orizzontalmente;
- **Titolo del documento**: nome esplicativo del contenuto del documento, centrato orizzontalmente e in grassetto;
- **Data**: data di approvazione finale dell'elaborato, centrata orizzontalmente;
- **Tabella informativa**: contenente informazioni quali:
  - versione del documento;
  - nominativo dell'approvatore (e quindi responsabile) del documento;
  - nominativo/i del/i redattore/i del documento;
  - nominativo/i del/i verificatore/i del documento;
  - stato del documento (approvato o non approvato);
  - tipo d'uso del documento (interno o esterno);

  - nominativi dei destinatari del documento;
  - email di riferimento del gruppo.

- **Descrizione**: breve riassunto del contenuto del documento.

3.1.5.3 Registro delle modifiche

La pagina successiva mostra una tabella contenente una raccolta cronologica delle modifiche avvenute sul documento. Le colonne espongono i seguenti contenuti:

- versione del documento dopo la modifica;
- data della modifica;
- nominativo del componente responsabile della modifica;
- ruolo del componente responsabile della modifica;
- breve descrizione della modifica effettuata;

3.1.5.4 Indice

L'indice riassume in modo schematico i contenuti del documento mostrando la loro suddivisione in sezioni, sottosezioni e paragrafi. È presente all'inizio di ogni elaborato dopo il registro delle modifiche. I suoi elementi sono cliccabili e rimandano alla pagina corrispondente. In alcuni documenti viene corredato di due liste:

- **Lista delle immagini**: riportante le immagini utilizzate tramite rispettiva didascalia;
- **Lista delle tabelle**: riportante le tabelle utilizzate tramite rispettiva didascalia (sono esclusi il registro delle modifiche e il riepilogo tracciamenti)

3.1.5.5 Contenuto

Il corpo principale del documento è così strutturato:

- intestazione in alto con logo del gruppo riadattato a sinistra e nome del documento a destra;
- una riga divisoria;

- il contenuto della pagina formattato con una dimensione di 12 punti e la visualizzazione in A4 formato articolo (`\documentclass{article}`);
- una riga divisoria opzionale in caso di appunti;
- eventuali note sul testo riportate con un numero progressivo a esponente;
- una riga divisoria;
- il contenuto a piè di pagina con a destra il numero progressivo della pagina visualizzata sul totale (nell'indice invece dei numeri arabi vengono utilizzati quelli romani).

3.1.6 Norme tipografiche

3.1.6.1 Nomi dei file

Per organizzare i file in modo più ordinato e corretto il gruppo ha scelto di seguire la pratica dello "snake_case"$_G$. Tale prassi, nell'ambito di questo progetto$_G$, è applicata seguendo tre regole:

- i file con nome composto da più parole useranno il carattere underscore come separatore;
- i nomi dei file saranno scritti interamente in minuscolo;
- non vengono tralasciate preposizioni di alcun tipo dal nome del file.

3.1.6.2 Glossario

- Ogni parola da inserire nel *Glossario* è corredata da una "G" a pedice per tutte le sue occorrenze;
- sono escluse dalla regola precedente i titoli e il nome delle sezioni del documento.

3.1.6.3 Stili di testo

- **Grassetto**: il grassetto viene usato solamente per evidenziare le parole a cui si vuole dare risalto e per la prima parola di un elenco puntato se quest'ultimo è del tipo: **Termine**: definizione/spiegazione.
- **Corsivo**: il corsivo viene usato per evidenziare i nomi dei documenti oltre che per il nome dell'azienda proponente (*Zucchetti*) e del gruppo (*VRAM Software*).

- **Maiuscolo**: in aggiunta alle normali regole dell'italiano le lettere maiuscole verranno utilizzate anche per riportare i nomi dei documenti e di certi capitoli.

3.1.6.4 Elenchi puntati

Ogni voce dell'elenco puntato sarà preceduta da un "•" (punto elenco) per quanto riguarda il primo livello e da un "-" per il secondo. Alla fine di ogni elemento, invece, si aggiunge il carattere ";" a meno che non sia l'ultimo dove va messo il segno di punteggiatura ".". Se la voce dell'elenco è del tipo termine-spiegazione/definizione il termine va in grassetto.

3.1.6.5 Convenzioni

Vengono usate tre convenzioni per la data, l'orario (i quali prendono come riferimento ISO 8601) e la valuta. Le date saranno espresse nella forma:

YYYY-MM-DD

dove:

- **YYYY**: indica l'anno;
- **MM**: indica il mese e deve sempre avere due cifre;
- **DD**: indica il giorno e deve sempre avere due cifre.

Gli orari saranno invece rappresentati nella forma:

HH:MM

dove:

- **HH**: indica le ore (in formato 24 ore) e deve sempre avere due cifre;
- **MM**: indica i minuti e deve sempre avere due cifre.

La valuta infine sarà scritta nella forma:

X.XXX,YY

dove:

- **XXXX**: rappresenta la parte intera della somma;
- **.**: viene usato come separatore ogni tre cifre intere;
- **YY**: rappresenta la parte decimale della somma;

3.1.7 Produzione

Il progetto$_G$ prevede la stesura di documenti che si dividono in interni ed esterni a seconda dei destinatari. Questi elaborati verranno poi valutati in quattro revisioni. Tali elementi progettuali saranno elencati e spiegati di seguito con le rispettive sigle.

3.1.7.1 Documenti esterni

- **Analisi dei Requisiti - AdR**: contiene la descrizione degli attori$_G$ del sistema e la loro interazione con i casi d'uso$_G$, fornendo una visione chiara ai progettisti sul problema da trattare;
- **Manuale Utente - MU**: serve per guidare l'utente nell'utilizzo del software;
- **Manuale Sviluppatore - MS**: ha lo scopo di spiegare a eventuali sviluppatori esterni il funzionamento del sistema;
- **Piano di Progetto - PdP**: dà indicazioni sulle tempistiche e i costi che il gruppo prevede d'impiegare per la realizzazione del progetto$_G$;
- **Piano di Qualifica - PdQ**: fornisce le specifiche riguardanti il controllo qualità dei processi$_G$ e dei prodotti$_G$ tramite delle metriche$_G$ misurabili.

3.1.7.2 Documenti interni

- **Glossario - G**: documento dove sono riportate i termini che necessitano di spiegazione o che potrebbero causare confusione o fraintendimenti;
- **Norme di Progetto - NdP**: linee guida per la gestione delle attività di progetto$_G$;
- **Studio di Fattibilità - SdF**: breve analisi di tutti i capitolati$_G$ proposti.

3.1.7.3 Verbale

Il verbale della riunione è un documento particolare in quanto può essere sia interno che esterno. Sarà interno quando i partecipanti saranno solo i componenti del gruppo; esterno quando saranno presenti anche referenti dell'azienda. È inoltre presente la tabella "Riepilogo dei tracciamenti" che tiene

segno delle modifiche discusse e approvate in sede d'incontro. Questo prospetto presenta una colonna con un codice nel formato:

V(I/E)_X.Y

dove:

- **V(I/E)**: indica se il verbale è interno o esterno;
- **X**: indica il numero dell'incontro che si sta organizzando;
- **Y**: indica il numero della scelta.

3.1.7.4 Revisioni

- **Revisione dei Requisiti - RR**: analisi iniziale del capitolato per concordare i requisiti col proponente;
- **Revisione di Progettazione - RP**: revisione intermedia per accertare la fattibilità del progetto$_G$;
- **Revisione di Qualifica - RQ**: revisione intermedia per l'approvazione delle verifiche e l'attivazione del processo$_G$ di validazione$_G$;
- **Revisione di Accettazione - RA**: revisione finale per il collaudo del software e l'accertamento del soddisfacimento dei requisiti presentati in RR.

3.1.7.5 Altre sigle

- **Responsabile - Re**;
- **Amministratore - Am**;
- **Analista - An**;
- **Progettista - Pr**;
- **Programmatore - Pt**;
- **Verificatore - Ve**;
- **Proof of Concept - PoC**$_G$.

3.1.8 Elementi grafici

3.1.8.1 Immagini

Le immagini sono sempre centrate e presentano una didascalia, tranne il logo nella prima pagina.

3.1.8.2 Tabelle

Le tabelle sono sempre centrate e presentano una didascalia, tranne il "Registro delle modifiche" e il "Riepilogo tracciamenti". Inoltre le tabelle presentano un'alternanza di colori per favorirne la lettura.

3.1.8.3 Didascalie

Le didascalie indicano in maniera progressiva il numero di tabella o immagine del documento e descrivono brevemente il loro contenuto.

3.1.9 Strumenti

3.1.9.1 LaTeX

Per stilare i documenti è stato usato il linguaggio di markup LaTeX. Questo linguaggio permette di scrivere dei testi mantenendo il focus sul contenuto e non sulla forma. La struttura infatti è stata dichiarata all'inizio del progetto$_G$ e poi riutilizzata permettendo una gestione degli elaborati scalabile, modulare e ordinata.

3.1.9.2 TeXStudio

Per scrivere in LaTeXviene usata l'IDE *TeXStudio* creata appositamente per questo fine e che quindi offre delle scorciatoie per i vari comandi, un correttore automatico per la lingua italiana e un servizio di compilazione e visualizzazione dei PDF integrato;

3.1.9.3 Lucidchart

Per la creazione dei diagrammi UML$_G$ e di altro tipo viene usata la piattaforma online Lucidchart che consente agli utenti di collaborare alla stesura, alla revisione e alla condivisione di grafici.

`https://www.lucidchart.com`

3.2 Gestione della configurazione

3.2.1 Descrizione

In questo capitolo verrà illustrato come il gruppo ha gestito la configurazione degli strumenti e delle risorse utilizzate per svolgere il progetto$_G$. Sarà quindi descritta la configurazione del repository$_G$ su GitHub, del sistema di versionamento$_G$ Git e dei servizi GitHub.

3.2.2 Previsioni

Questa sezione ha lo scopo supportare i processi di documentazione, di sviluppo e manutenzione del software rendendoli definiti e ripetibili.

3.2.3 Obiettivi

L'obiettivo principale è quindi di rendere documenti e codice sorgente univocamente identificati e facilmente identificabili, evidenziandone versioni e modifiche. Vuole anche agevolare l'identificazione delle relazioni esistenti fra gli elementi e fa da supporto alla fase di verifica.

3.2.4 Versionamento

Le modifiche effettuate ai documenti ed ai file contenenti codice sorgente sono identificate da un numero di versione presente all'interno dei file stessi. Per quanto riguarda i documenti, ogni numero di versione sarà presente nel nome del file ed avrà una corrispondente riga nella tabella delle modifiche così da avere uno storico delle modifiche effettuate al documento. Per il codice sorgente del software le modifiche effettuate in una versione saranno consultabili sulla sezione rilasci di GitHub e saranno identificate da opportuni tag.
Per identificare univocamente le versioni dei documenti seguiamo le indicazioni dell'ente ETSI, dunque la numerazione delle versioni è composta da 3 cifre nel formato X.Y.Z e la prima bozza ha versione 0.0.1. Le modifiche al documento aggiorneranno le cifre nel seguente modo:

- **Z** : Questa cifra viene incrementata ogni volta che vengono effettuate delle modifiche editoriali$_G$ al documento, es. X.Y.1, X.Y.2 etc.;
- **Y** : Questa cifra viene incrementata ogni volta che vengono effettuate delle modifiche tecniche$_G$ al documento. Se sono state eseguite modifiche sia editoriali$_G$ che tecniche$_G$ allora entrambe le cifre Z e Y saranno incrementate, es. X.1.1, X.2.2 etc;

- **X** : Questa cifra viene incrementata ogni volta che viene rilasciata una nuova versione finale del documento, ovvero una versione di rilascio.

La prima versione finale ha versione 1.1.1.
Durante la modifica e la revisione del documento le bozze vengono incrementate le cifre Z ed Y, es. 1.2.0, 1.2.1, 1.3.0, etc.
Quando il documento viene approvato come finale allora viene incrementata la cifra X, ad esempio la bozza 1.5.3 diventa la versione finale 2.1.1.
Ogni incremento sulle singole cifre è rigorosamente un +1, non possono essere saltati numeri.

3.2.5 Gestione delle modifiche

Al fine di monitorare e limitare le modifiche al ramo principale del repository$_G$, master, è utilizzato il meccanismo di pull request fornito da GitHub. Ogni membro del gruppo può creare branch secondari, secondo il workflow$_G$ feature branch, su cui effettuare modifiche, tuttavia per unirle al branch master è necessario aprire una pull request che dovrà essere revisionata dai verificatori tramite i servizi di revisione integrati in GitHub. Una volta revisionata positivamente è compito del responsabile del documento approvare la pull request ed effettuare quindi l'effettiva unione delle modifiche nel branch master.
In sintesi, per effettuare modifiche ai documenti sono previsti i seguenti passaggi:

- Contattare il responsabile del documento affinché autorizzi la modifica del documento;
- Creare un branch secondario ed effettuare le modifiche al documento;
- Aprire una pull request per unire le modifiche al ramo master;
- I verificatori revisionano la pull request ed eventualmente richiedono aggiornamenti;
- Completata la revisione il responsabile approva la pull request.

3.2.6 Repository

Per tenere traccia di versioni e modifiche fatte a documenti e codice è utilizzato il sistema di versionamento$_G$ distribuito Git, che può essere utilizzato tramite riga di comando o utilità grafiche come GitHub Desktop o GitKraken.
Il repository$_G$ dei documenti è ospitato sul sito GitHub all'indirizzo:

`https://github.com/marcoDallas/VRAM_SW_DOC_2020/`

Al fine di fornire una navigazione agevole e standardizzata, il contenuto del repository$_G$ è organizzato in modo gerarchico tramite directory secondo il seguente schema:

```
root
├── RR
│   ├── Esterni
│   └── Interni
├── RP
│   ├── Esterni
│   └── Interni
├── RQ
│   ├── Esterni
│   └── Interni
├── RA
│   ├── Esterni
│   └── Interni
├── Guide
└── Template
    ├── Images
    ├── config
    ├── img
    └── res
```

Nel dettaglio:

- **RR** : contiene i sorgenti LaTeX dei documenti relativi alla revisione dei requisiti;
- **RP** : contiene i sorgenti LaTeX dei documenti relativi alla revisione di progettazione$_G$;
- **RQ** : contiene i sorgenti LaTeX dei documenti relativi alla revisione di qualifica;
- **RA** : contiene i sorgenti LaTeX dei documenti relativi alla revisione di accettazione;
- **Guide** : contiene brevi indicazioni interne su come usare i comandi LaTeX usati nei documenti e su come riutilizzare il template per generare nuovi documenti;

- **Template** : contiene i sorgenti LaTeX usati per generare la base comune di tutti i documenti.

È inoltre disponibile una directory condivisa su Google Drive, con la stessa struttura gerarchica sopra esposta, che contiene tutti e soli i file PDF in versione finale. Lo scopo di questa directory condivisa è di facilitare la consultazione e la condivisione dei file PDF stessi.

3.2.7 Tipologie di file non accettate

Tramite un apposito file .gitignore presente nella root directory della gerarchia vengono definiti i tipi di file non accettati all'interno del repository$_G$. Vengono esclusi tutti i file di compilazione, compilati o temporanei in quanto il repository$_G$ dovrebbe contenere solamente i seguenti formati di file:

- File sorgente LaTeX con estensione .tex;
- File immagine, preferibilmente in formato .png;
- File testuali in formato .md o .txt.

3.2.8 Ulteriori Strumenti GitHub utilizzati

In aggiunta ai servizi già elencati, al fine di migliorare efficacia ed efficienza, vengono utilizzate le funzionalità di "Issue Tracking System", "Milestone" e "Project Board" integrate in GitHub. Ognuna di queste funzionalità viene usata solo da chi autorizzato, ad esempio rilasci di versioni, creazione e chiusura di milestone sono concesse solo al responsabile di progetto$_G$.
Per il repository$_G$ contenente il codice sorgente del software saranno usate anche le GitHub Actions al fine di implementare la pratica della continuous integration.

3.3 Gestione della qualità

3.3.1 Scopo

Lo scopo della gestione della qualità è quello di dare una valutazione oggettiva riguardo ai servizi che il prodotto$_G$ deve fornire in modo da mantenere uno standard di qualità prestabilito e da soddisfare i requisiti impliciti ed espliciti del proponente.

3.3.2 Aspettative

Le aspettative della gestione della qualità possono essere riassunti dai seguenti punti, che rappresentano gli obiettivi che vogliono essere raggiunti dal gruppo. Si vuole ottenere:

- Qualità per tutta la durata del ciclo di vita del software;
- Una copertura totale di tutti i requisiti impliciti ed espliciti richiesti dal proponente;
- La soddisfazione del cliente con il prodotto$_G$ sviluppato;
- Oggettività nella ricerca della qualità in modo da ottenerla in tutti i processi e in tutti i prodotti$_G$.

3.3.3 Descrizione

Per la gestione della qualità dello sviluppo di questo progetto$_G$ si è cercato di applicare lo standard ISO 9000, il quale afferma che il sistema di qualità, cioè l'insieme dei metodi e regole per gestire la qualità di un prodotto$_G$, è costituito da tre passaggi: lo sviluppo, il controllo e il miglioramento continuo. Questi tre passaggi sono ripetuti per ogni processo$_G$ in modo di cercare di prevenire possibili errori e anomalie. Per una descrizione approfondita sul piano della qualità, che è uno dei costituenti del sistema di qualità, si faccia riferimento al seguente documento: *Piano di Qualità v. 1.1.1*. Nel documento sopracitato sono descritte le metodologie e regole utilizzate dal gruppo *VRAM Software* per cercare di raggiungere le aspettative definite nella sezione precedente. Il contenuto sintetico del documento *Piano di Qualità v. 1.1.1* contiene:

- la presentazione delle caratteristiche e gli attributi considerati più importanti del prodotto$_G$;
- la descrizione degli standard di qualità utilizzati per la gestione del ciclo di vita del software;

- l'individuazione dei processi$_G$ definiti dagli standard descritti, nei quali sono successivamente stati individuati gli obiettivi, le strategie per raggiungerli e le metriche$_G$ per misurarne l'avanzamento.

Con il controllo e il miglioramento dei processi$_G$ a ogni prodotto$_G$ vengono poi associati: gli obiettivi da raggiungere e le metriche$_G$ utilizzate. L'obiettivo che si vuole raggiungere, utilizzando le regole proposte nel documento *Piano di Qualità v. 1.1.1*, è quello di ottenere un livello di qualità soddisfacente per software e documentazione, perché se si riesce a perseguire la qualità per tutta il ciclo di vita del prodotto$_G$, allora le possibilità di ottenere un prodotto$_G$ stabile, fonte di miglioramento continuo, sarà più alta.

3.3.4 Attività

Per ogni processo$_G$ abbiamo deciso di pianificare lo sviluppo in tre attività:

- **Pianificazione**
  - Ci siamo posti degli obiettivi per perseguire la qualità;
  - Abbiamo definito delle strategie per raggiungere tali obiettivi;
  - Abbiamo disposto le persone e le risorse da utilizzare per la realizzazione del processo$_G$.
- **Valutazione**
  Abbiamo messo in atto le strategie proposte nella pianificazione valutandone i risultati;
- **Reazione**
  Per ogni risultato ottenuto abbiamo modificato opportunamente le nostre strategie.

3.3.5 Strumenti

Gli strumenti utilizzati per la gestione della qualità sono:

- Strumenti forniti dallo standard ISO-12207, che contiene la descrizione del processo$_G$: "Gestione della Qualità";
- Le metriche$_G$ utilizzate per misurare il perseguimento della qualità.

3.4 Verifica

3.4.1 Scopo

La verifica ha lo scopo di accertare che i risultati ottenuti da tutte le attività, attuate in un periodo preso in considerazione, raggiungano tutti i requisiti richiesti senza aver introdotto degli errori. La verifica deve essere svolta durante il ciclo di vita del software, più precisamente al passare di ogni baseline$_G$.

3.4.2 Aspettative

La verifica viene effettuata per raggiungere i seguenti obiettivi:

- Cercare consistenza, completezza e correttezza nelle singole attività;
- Ottenere supporto per una successiva validazione$_G$ del prodotto$_G$;
- Avere a disposizione dei criteri e procedure, comprensibili e affidabili, da seguire.

3.4.3 Descrizione

Il processo$_G$ di verifica assicura che tutte le attività svolte della fase in esame, attraverso analisi e test successivamente definiti, siano conformi alle aspettative.

3.4.4 Attività

3.4.4.1 Analisi

Le attività che si effettuano durante la verifica sono due tipi di analisi: l'analisi statica$_G$ e l'analisi dinamica.

Analisi statica

L'analisi statica$_G$ è lo studio della documentazione e del codice sorgente, essa non richiede l'esecuzione delle singole parti del sistema software che si sta sviluppando, quindi viene applicata ad ogni prodotto$_G$ di processo$_G$. L'analisi statica$_G$ può essere effettuata manualmente in due modi:

- **Walkthrough**
  Metodo di lettura in cui tutti i componenti del gruppo analizzano tutti i documenti e codice realizzato, senza tralasciare nulla, in modo da rilevare eventuali errori. Questa ricerca è ad ampio spettro, infatti viene effettuata anche se a priori non si può sapere se verranno trovate anomalie;

- **Inspection**
  Metodo di lettura o desk check per cercare errori specifici utilizzando liste di controllo o check list, la tabella successiva rappresenta gli errori che i nostri verificatori hanno identificato come essere i più frequenti.

Oggetto da controllare	Correzione
Formato data dei documenti	Tutte le date devono utilizzare la seguente scrittura: *YYYY-MM-DD*
Sintassi nome documenti citati	I documenti, interni o esterni, citati nella documentazione devono essere scritti in italico utilizzando il comando: `\textit{}`
Sintassi parole da enfatizzare	Le parole da enfatizzare, come titolo di liste puntate, devono essere scritte i grassetto utilizzando il comando: `\textbf{}`
Punteggiatura liste puntate	Ogni elemento nelle liste puntate deve iniziare con una lettera maiuscola e deve finire con un punto e virgola
Uso corretto dei comandi `\glo` e `\glosp`	I comandi `\glo` e `\glosp`, sono stati creati per identificare le parole da inserire nel glossario, `\glo` dovrà essere usato se la parola del glossario precede un carattere di punteggiatura. Il comando `\glosp` sarà invece usato per le parole che precedono uno spazio

Tabella 1: Tabella errori più frequenti

Analisi dinamica

L'analisi dinamica è una tecnica che consiste nell'esecuzione del prodotto$_G$

software eseguendo dei test su un insieme finito di casi.

3.4.4.2 Test

Per la corretta verifica del codice, i test devono seguire regole ben precise che definiscono i parametri e le proprietà che devono avere test per essere considerati efficaci:

Parametri da definire:

- **Input**: l'input preso in considerazione;
- **Output**: l'output che si aspetta di ricevere;
- **Ambiente**: l'ambiente, quindi hardware e sistema operativo, in cui viene eseguito il test;
- **Stato iniziale**: lo stato iniziale del prodotto$_G$, precedente all'esecuzione del test;
- **Istruzioni opzionali**: l'utilizzo di istruzioni opzionali aggiuntive deve essere noto.

Un test può essere definito efficace se:

- I parametri citati sopra sono correttamente indicati;
- Il nome di un test deve essere autoesplicativo per capire subito che cosa si sta testando;
- Esso è ripetibile, quindi ci si aspetta un risultato costante per tutte le esecuzioni di quel test;
- In caso di fallimento del test, esso devo fornire informazioni utili a risolvere il problema identificato;
- In caso di possibili effetti indesiderati ci si aspetta un avvertimento;
- Esso è esaustivo e accurato, infatti devono poter identificare una parte del progetto$_G$ che potrebbe essere causa di errori.

Il cosiddetto software testing è un procedimento importante per misurare la qualità di un prodotto$_G$ software, per questo motivo sono definiti cinque livelli che dividono i tipi di test da effettuare secondo una gerarchia. Essi sono:

- **Test d'unità**;

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

Per ogni test vengono definiti lo stato e l'esito:

- Stato:
  - I: implementato;
  - NI: non implementato.
- Esito:
  - P: positivo;
  - N: negativo;
  - NE: non eseguito.

Test d'unità

I test d'unità si definiscono tali perché controllano il corretto funzionamento di una sezione di codice specifica chiamata unità. Usualmente questi tipi di test vengono eseguiti su funzioni o, se si parla di programmazione ad oggetti, su un metodo di una classe, quindi su una parte minimale del codice. Il loro scopo è verificare che gli input forniti diano gli output attesi. È buona norma rendere questi test automatici per minimizzare il tempo di esecuzione senza richiedere l'interazione dello sviluppatore. Questo tipo di test viene scritto dal programmatore che sviluppa le singole unità mediate l'ausilio di driver$_G$ e stub$_G$, per verificare l'assenza di errori e documentare il funzionamento dell'unità.

Test d'integrazione

Successivamente ai test d'unità, si assemblano gruppi di unità progressivamente che formeranno degli agglomerati sempre più grandi. Questo tipo di test, infatti, si concentra nella ricerca di anomalie ed errori tra le interfacce dei singoli componenti. Una volta concluso un singolo test, si otterrà un agglomerato che costituirà una nuova unità da cui partire per ulteriori test d'integrazione. Per svolgere questo test si è scelto un approccio iterativo.

Test di sistema

I test di sistema controllano il funzionamento corretto del prodotto$_G$ e verificano che il sistema soddisfi tutti i requisiti definiti nel documento: *Analisi dei Requisiti*. Con l'esecuzione di questi test si riesce a definire e documentare tutte le funzionalità del sistema e possono individuare criticità come il problema del black-box$_G$. Il gruppo svolgerà questo test quando si raggiungerà uno stato di rilascio per il prodotto$_G$ sviluppato. Per identificare un test di unità viene utilizzata una codifica con il seguente formato: **TS[codice]** Dove codice è un numero progressivo nel formato: [X].[Y] con X e Y numeri maggiori di zero.

Test di accettazione

I test di accettazione, o di collaudo, vengono effettuati ad ogni pre-rilascio, quindi dopo l'esecuzione dei test di sistema. Essi servono a confermare il soddisfacimento dei requisiti da parte del committente; infatti questi tipi di test ne richiedono la presenza. Se questi test vengono superati significa che il prodotto$_G$ software è pronto per il rilascio. Per identificare un test di accettazione viene utilizzata una codifica con il seguente formato: **TA[codice]** Dove codice è un numero progressivo nel formato: [X].[Y] con X e Y numeri maggiori di zero.

3.4.5 Strumenti

3.4.5.1 Correzione ortografica

Il nostro gruppo ha utilizzato, per la ricerca di errori ortografici o stilistici nel testo dei documenti, la libreria HunSpell utilizzata con l'editor di test: TexStudio. Esso permette di avere un controllo in tempo reale su errori ortografici in lingua inglese e italiana (sottolineati in rosso) e ripetizioni di parole a breve distanza (sottolineati in verde).

3.5 Validazione

3.5.1 Scopo

Lo scopo del processo$_G$ di validazione è accertare che il prodotto$_G$, per uno specifico uso, corrisponda alle attese.

3.5.2 Descrizione

Le attese sono quelle del committente e corrispondono, se stilata correttamente, all'analisi dei requisiti. Per eseguire una validazione si deve avere un prodotto$_G$ che è a un sufficiente livello di avanzamento, quindi il numero di validazioni svolte è molto minore rispetto al numero delle verifiche, che iniziano a venire eseguite prima.

3.5.3 Attività

Ci sono due modi di eseguire una validazione: internamente o esternamente. Una validazione interna ha successo se i requisiti in precedenza scritti sono soddisfatti, una esterna invece se il committente dà parere positivo. Più nel particolare la validazione consiste nel:

- Selezionare i requisiti da soddisfare;
- Individuare dei test specifici per misurare il soddisfacimento dei requisiti selezionati;
- Eseguire i test;
- Analizzare i risultati dei test.

4 Processi organizzativi

4.1 Gestione organizzativa

4.1.1 Scopo

Lo scopo della gestione organizzativa è composto dai seguenti punti:

- Creazione di un modello organizzativo che specifica i rischi che possono verificarsi;
- Definizione un modello di sviluppo da seguire;
- Pianificazione del lavoro in base alle scadenze fissate;
- Creazione e calcolo di un piano economico suddiviso tra i ruoli;
- Definizione finale del bilancio sul totale delle spese.

Queste attività sono definite a cura del responsabile di progetto$_G$ e redatte all'interno del documento *Piano di Progetto*.

4.1.2 Aspettative

Gli obiettivi della gestione organizzativa sono:

- Coordinare i componenti del gruppo attraverso l'assegnazione di ruoli e compiti;
- Coordinare la comunicazione tra i membri del gruppo con l'ausilio di strumenti che la rendano facile ed efficace;
- Pianificare l'esecuzione delle attività in modo ragionevole;
- Gestire le attività anche dal punto di vista economico;
- Monitorare costantemente il team, i processi$_G$ e i prodotti$_G$ in modo efficace.

4.1.3 Descrizione

Il processo$_G$ di gestione organizzativa è composto dalle seguenti attività:

- Definizione dello scopo;
- Generazione delle istanze dei processi$_G$;

- Stima e pianificazione di risorse, costi e tempo per lo svolgimento del progetto$_G$;
- Assegnazione dei ruoli e, per ognuno di essi, delle attività;
- Gestione del controllo e dell'esecuzione di tutte le attività;
- Valutazione periodica dello stato e dell'esecuzione delle attività rispetto a quanto pianificato.

4.1.4 Ruoli di progetto

4.1.4.1 Assegnazione

I ruoli scelti corrispondono alla rispettiva figura aziendale e valgono per la durata di una milestone. Al termine di ognuna di esse, viene eseguita una rotazione dei ruoli così da permettere a ciascuno dei membri del gruppo di ricoprirli tutti almeno una volta. I ruoli previsti sono descritti di seguito.

4.1.4.2 Responsabile di progetto

La figura del responsabile di progetto$_G$ è molto importante in quanto convergono su di esso le responsabilità di gestione, pianificazione, coordinamento e controllo dell'intero progetto$_G$. Le sue mansioni sono:

- Controllo e coordinamento di risorse, componenti e attività del gruppo;
- Relazione con il controllo di qualità interno al progetto$_G$;
- Analisi e gestione degli elementi più critici e i rischi;
- Intermediazione nelle comunicazioni esterne ovvero con proponente e committente;
- Elaborazione ed emissione di piani e scadenza;
- Approvazione finale dei documenti;

4.1.4.3 Amministratore di progetto

La figura dell'amministratore è responsabile dell'efficienza e dell'operatività dell'ambiente di sviluppo e della redazione e attuazione di piani e procedure di gestione per la qualità. Le sue mansioni sono:

- Controllo versioni e configurazioni del prodotto$_G$;
- Gestione di tutta la documentazione del progetto$_G$;

- Direzione delle infrastrutture di supporto;
- Risoluzione del problemi sorti dalla gestione dei processi$_G$;
- Stesura del documento *Norme di Progetto* per conto del responsabile di progetto$_G$;
- Collaborazione con il responsabile di progetto$_G$alla reazione del documento *Piano di Progetto*.

4.1.4.4 Analista

La figura dell'analista è responsabile di tutta l'attività di analisi dei problemi e del dominio applicativo. Le sue mansioni sono:

- Studio del dominio del problema;
- Analisi della complessità, della fattibilità e dei requisiti del problema;
- Stesura del documento *Studio di Fattibilità*;
- Stesura del documento *Analisi dei Requisiti*.

4.1.4.5 Progettista

La figura del progettista è responsabile delle attività di progettazione$_G$. Deve perciò gestire gli aspetti tecnologici e tecnici del progetto$_G$. Le sue mansioni sono:

- Redigere le specifiche tecniche del prodotto$_G$ effettuando scelte il più efficienti e performanti possibili, sulla base delle tecnologie presenti;
- Sviluppare l'architettura del prodotto$_G$ tale da fornire una base stabile a manutenibile che soddisfi le specifiche tecniche.

4.1.4.6 Programmatore

La figura del programmatore è responsabile dell'attività di codifica mirata alla realizzazione del prodotto$_G$ compreso delle componenti necessarie allo svolgimento di test di verifica e validazione$_G$. Le sue mansioni sono:

- Eseguire l'attività di codifica sulla base del lavoro svolto da progettista;
- Fornire dei componenti che permettano si svolgere test e validazione$_G$ sul prodotto$_G$.

4.1.4.7 Verificatore

La figura del verificatore è responsabile di tutta l'attività di verifica. Si affida al documento *Norme di Progetto* è definito anche lo standard da seguire per questa attività. Le sue mansioni sono:

- Redazione della parte retrospettiva del piano di qualifica;
- Ispezione dei documenti al fine di controllare che siano conformi allo standard previsto nelle *Norme di Progetto*;
- Segnalazione di eventuali errori nelle parti di prodotto$_G$ a chi ha responsabilità su di esse.

4.1.5 Gestione delle comunicazioni

Comunicazioni interne Le comunicazioni interne al gruppo avvengono tramite la piattaforma Slack nella quale è stato creato un workspace suddiviso nei seguenti canali tematici:

- **Documentazione**: canale dedicato alle discussioni in merito alle attività di stesura e verifica della documentazione;
- **Sviluppo**: canale dedicato alle discussioni in merito alle attività di analisi, progettazione$_G$ e codifica del prodotto$_G$;
- **Generale**: canale dedicato alle comunicazioni di servizio per il coordinamento di attività, incontri e impegni del gruppo.

Nei canali dedicati alla documentazione e allo sviluppo è stata attivata l'integrazione con GitHub per segnalare i nuovi commit e le pull request con un messaggio. Inoltre è stata eseguita l'integrazione con Google Calendar che permette di avere una sezione dedicata alle notifiche degli eventi programmati sul calendario e l'integrazione del plug-in Simple Poll che permette di eseguire dei sondaggi per prendere le decisioni.

Comunicazioni esterne Le comunicazioni esterne avvengono tramite la posta elettronica all'indirizzo `vram.software@gmail.com`. Il responsabile di progetto$_G$ ha l'incarico di gestire queste comunicazioni ed è tenuto ad informare tutti gli elementi del gruppo degli argomenti trattati qualora non fossero presenti.

4.1.6 Gestione degli incontri

Incontri interni Gli incontri del gruppo sono accordati su Slack nel canale generale oppure nella precedente riunione e vengono annotati su Google Calendar. Essi sono identificati da:

- Data;
- Ora di inizio;
- Durata prevista;
- Elenco di attività da svolgere.

Verbali di incontri interni Un segretario, nominato dal responsabile, è incaricato della redazione di un *Verbale* della riunione che tiene traccia di:

- **Luogo**;
- **Data**;
- **Ora d'inizio**;
- **Ora di fine**;
- **Partecipanti**;
- **Argomenti trattati**: argomenti all'ordine del giorno che sono trattati nell'incontro da verbalizzare;
- **Discussioni e decisioni prese**.

Incontri esterni Gli incontri esterni vengono concordati se i membri del gruppo, il committente o il proponente lo ritengono necessario. In tal caso, sarà compito del responsabile di progetto$_G$ definire data e ora dell'incontro in accordo tra le due parti attraverso gli appositi canali di comunicazione descritti precedentemente.

Verbali di incontri esterni Un segretario, nominato dal responsabile, è incaricato della redazione di un *Verbale* della riunione tenendo traccia di:

- **Luogo**;
- **Data**;
- **Ora d'inizio**;

- **Ora di fine**;
- **Partecipanti**;
- **Argomenti trattati**: argomenti all'ordine del giorno che sono trattati nell'incontro da verbalizzare;
- **Verbale**: esposizione delle discussioni e delle decisioni prese.

4.1.7 Gestione degli strumenti di coordinamento

Ticketing Il sistema di ticketing permette ai membri del gruppo di gestire le tutte le attività del progetto$_G$. Per eseguire il ticketing vengono utilizzati gli strumenti dell'Issue Tracking System presenti gratuitamente in GitHub ed il funzionamento è il seguente: Il responsabile di progetto$_G$ ha l'incarico di assegnare i compiti ai membri del gruppo attraverso le Issue. Ognuna di esse contiene titolo, descrizione, milestone$_G$, le project board Kanban automatizzate associate e il destinatario. Le project board Kanban automatizzate permettono al responsabile di progetto$_G$ di monitorare l'andamento delle attività tracciando lo stato di ogni singolo compito che può essere:

- To Do (da fare);
- In Progress (in lavorazione);
- Done (concluso).

L'associazione con le milestone$_G$ definisce le scadenze delle singole Issue. Inoltre è presente una comoda funzionalità di ricerca per le Issue assegnate che ne agevola ulteriormente la gestione.

4.1.8 Gestione dei rischi

La gestione e l'analisi dei rischi è un'attività a carico del responsabile di progetto$_G$ che deve essere documentata nel documento *Piano di Progetto v. 1.1.1*. La procedura per la gestione dei rischi è composta dai seguenti passi:

- Individuazione dei rischi;
- Analisi dei rischi;
- Pianificazione dell'attività di controllo;
- Controllo e monitoraggio.

Viene definita una codifica per l'identificazione dei fattori di rischio: R[tipo][X]

- tipo:
  - **RT**: rischi tecnologici di lavoro e produzione del software;
  - **RG**: rischi di gruppo;
  - **RO**: rischi organizzativi delle attività da svolgere;
  - **RR**: rischi legati ai requisiti e ai rapporti con gli stakeholder$_G$;
  - **RS**: rischi di stima di costi e tempi.
- X: numero intero progressivo che inizia da 1.

Inoltre vengono definiti delle misure di probabilità e degli indici di gravità:

- Probabilità:
  - **Alta**;
  - **Media**;
  - **Bassa**.
- Gravità:
  - **1**: basso livello di gravità;
  - **2**: medio livello di gravità;
  - **3**: alto livello di gravità.

4.1.9 Metriche di qualità

4.1.9.1 Pianificazione delle risorse

I parametri utilizzati per valutare la qualità della pianificazione sono:

- **BAC: Budget at Completion** che rappresenta il budget totale del progetto$_G$ e si misura col relativo numero intero;
- **PV: Planned Value** che rappresenta il valore dei lavoro pianificato fino a quel momento e si misura con la formula: BAC $\cdot$ %lavoro pianificato;
- **EV: Earned Value** che rappresenta il valore del lavoro svolto fino a quel momento e si misura con la formula: BAC $\cdot$ %lavoro pianificato;
- **AC: Actual cost** che rappresenta il denaro speso fino a quel momento e si misura col relativo numero intero;

- **CPI: Cost Performance Index** che rappresenta un indice per i costi effettivi rispetto a quelli previsti e si misura con la formula: $\frac{EV}{AC}$, se è $\leq$1 indica che si sta spendendo più di quanto preventivato;
- **SPI: Schedule Performance Index** che rappresenta un indice per i tempi effettivi rispetto a quelli previsti e si misura con la formula: $\frac{EV}{PV}$, se è $\leq$1 indica che si sta impiegando più tempo di quanto preventivato;
- **EAC: Estimated Cost at Completion** che rappresenta il budget totale stimato del progetto$_G$ con il CPI del momento e si misura con la formula: $AC + \frac{BT-EV}{CPI}$;
- **SAC: Schedule at Completion** che rappresenta il tempo totale stimato del progetto$_G$ con SPI del momento e si misura con la formula: $\frac{TT}{SPI}$; con TT=Tempo Totale.

4.1.10 Strumenti

Gli strumenti utilizzati sono i seguenti:

- **Telegram**: strumento di messaggistica istantanea utilizzato inizialmente dal gruppo per il primo incontro;
- **Slack**: strumento utilizzato per le comunicazione interne del gruppo;
- **Google Drive**: strumento utilizzato per file che non sono oggetto di cambiamenti frequenti come il *Glossario* o immagini di vario tipo;
- **Google Calendar**: strumento utilizzato per tracciare e semplificare la gestione degli impegni e fornire a tutti notifiche e promemoria per gli stessi;
- **Gmail**: strumento di posta elettronica utilizzato per le comunicazioni esterne.
- **Skype**: strumento utilizzato per effettuare chiamate VoIP per alcuni incontri interni;
- **Git**: sistema di controllo di versionamento$_G$;
- **GitKraken, GitHub Desktop**: interfacce l'utilizzo di GitHub in locale;
- **GitHub**: strumento utilizzato per il proprio Issue Tracking System e sistema di versionamento$_G$ dei file.

4.2 Formazione del gruppo

4.2.1 Scopo

Lo scopo di questo processo$_G$ è formare ogni membro del gruppo per garantire che ognuno abbia le conoscenze e le competenze adeguate allo svolgimento dei compiti assegnati.

4.2.2 Descrizione

Il processo$_G$ di formazione è composto da un insieme di attività che definiscono come vengono formati i componenti del gruppo. Lo studio necessario per svolgere i compiti e utilizzare gli strumenti viene principalmente eseguito in modo individuale e autonomo da un componente del gruppo definito periodicamente dal responsabile di progetto$_G$. Al termine dello studio, egli deve comunicare ciò che ha imparato agli altri membri per garantire che tutti apprendano le nozioni. Inoltre, qualora un componente ritenga di non avere ancora sufficienti capacità o conoscenze per svolgere un compito assegnato, si deve rivolgere al responsabile di progetto$_G$ che ha l'incarico di organizzare un seminario mirato per il suo apprendimento.

4.2.3 Condivisione del materiale

Il materiale del progetto$_G$ è inserito all'interno di un sistema di versionamento$_G$ che permette ai membri del gruppo di integrare le proprie conoscenze e apprendere dal lavoro altrui. Per ogni documento, il materiale utilizzato per la comprensione dei concetti e la stesura è indicato nei "Riferimenti Informativi". Inoltre sono presenti i "Riferimenti Normativi" dentro i quali è presente il documento *Norme di Progetto v. 1.1.1* che tutti devono rispettare.

A Standard di qualità

A.1 ISO/IEC 25010

Lo standard internazionale ISO/IEC 25010 è utilizzato per misurare la qualità del software. Per farlo utilizza una serie di parametri e caratteristiche che possono essere utilizzate per valutare la qualità del software, grazie alle metriche fornite dallo standard ISO/IEC 25023.

A.1.1 Metriche di qualità interna

La qualità interna del software è quella relativa alle attività di progettazione e codifica, essa rappresenta la qualità della struttura e del codice del prodotto. La misurazione della qualità interna viene effettuata principalmente attraverso l'analisi statica del codice. Questi controlli sono utili perché semplificano il lavoro dei programmatori e permettono di apportare miglioramenti strutturali al software senza la necessità di eseguirlo. Con un buon livello di qualità interna si ottiene una solida struttura del prodotto, che sarà la base per la qualità esterna e la qualità in uso.

A.1.2 Metriche di qualità esterna

La qualità esterna del software è quella relativa a proprietà dinamiche e comportamentali del software. Essa viene infatti misurata attraverso l'analisi dinamica, ovvero tramite l'esecuzione del codice e lo svolgimento di test.

A.1.3 Metriche di qualità in uso

La qualità in uso è probabilmente la più difficile da misurare poiché è valutata in un ambito di utilizzo reale, con la partecipazione degli utenti. Essa dipende dalla qualità interna e da quella esterna; infatti per avere un livello di accettabile di qualità in uso è necessario un buon livello di qualità interna ed esterna.

A.1.4 Modello di qualità del prodotto

Per descrivere il modello di qualità del prodotto$_G$ software il nostro gruppo ha deciso di utilizzare lo standard ISO/IEC 25010, che determina le caratteristiche della qualità di un prodotto$_G$ software che verranno valutate. Il modello di qualità del prodotto$_G$, definito nello standard, è suddiviso nelle otto seguenti caratteristiche:

A.1.4.1 Idoneità funzionale

Per adeguatezza funzionale si intende la capacità di un prodotto$_G$ software di fornire funzioni che soddisfano i requisiti impliciti ed espliciti, quando usati in un determinato contesto sotto specifiche condizioni.

- **Completezza**: la capacità del software di offrire un insieme di funzioni le cui funzionalità coprono tutte le attività e tutti gli obiettivi di un utente;
- **Correttezza**: la capacità del software di fornire agli utenti risultati corretti e precisi rispetto al grado di precisione richiesto;
- **Adeguatezza**: la capacità del software di fornire funzioni appropriate che facilitano l'utente a raggiungere i suoi obiettivi.

A.1.4.2 Efficienza prestazionale

L'efficienza prestazionale è la caratteristica relativa alle prestazioni rispetto al numero di risorse usate sotto condizioni precise.

- **Comportamento rispetto al tempo**: la capacità del software di eseguire le funzioni in un tempo di risposta, tempo processionale e un volume di produzione che soddisfa i requisiti;
- **Utilizzo delle risorse**: la capacità del software di eseguire le funzioni usando un numero di risorse disponibili che soddisfa i requisiti;
- **Capacità**: la capacità del software di avere limiti prestazionali massimi che soddisfano i requisiti.

A.1.4.3 Compatibilità

La compatibilità è la caratteristica di un software di scambiare informazioni con altri prodotti$_G$, ed eseguire le sue funzioni mentre viene condiviso lo stesso ambiente hardware o software.

- **Coesistenza**: la capacità del software di eseguire le sue funzionalità senza impatti negativi mentre condivide lo stesso ambiente con un altro software;
- **Interoperabilità**: la capacità di due o più software di scambiare informazioni tra loro e usare tali dati per i loro scopi.

A.1.4.4 Usabilità

L'usabilità è la caratteristica di un software facilmente utilizzabile, comprensibile dall'utente ed efficacie a raggiungere gli obiettivi che un utente si prefissa.

- **Appropriatezza-Riconoscibilità**: la capacità del software di essere riconosciuto dall'utente come un prodotto$_G$ adatto ai loro scopi;
- **Apprendibilità**: la capacità del software di ridurre l'impegno che un utente deve avere per capire le funzionalità del prodotto$_G$;
- **Operabilità**: la capacità del software di essere utilizzato dagli utenti in modo semplice;
- **Protezione errori dell'utente**: la capacità del software di proteggere l'utente dagli errori che lo stesso può causare;
- **Estetica dell'interfaccia utente**: la capacità del software di avere una interfaccia utente piacevole;
- **Accessibilità**: la capacità del software di essere usato da persone con caratteristiche che alterano in modo negativo l'operabilità del software.

A.1.4.5 Affidabilità

Un prodotto$_G$ software è considerato affidabile quando mantiene un livello di prestazioni costanti per un determinato intervallo di tempo sotto precise condizioni che possono variare nel tempo.

- **Maturità**: la capacità del software di avere un comportamento affidabile quando usato in condizioni normali;
- **Disponibilità**: la capacità del software di essere disponibile quando ne è richiesto l'uso;
- **Tolleranza agli errori**: la capacità del software di tollerare situazioni critiche in caso di errori, in modo da permettere funzionalità costanti anche in presenza di malfunzionamenti;

- **Recuperabilità**: la capacità del software di ristabilire il sistema allo stato desiderato dopo malfunzionamenti.

A.1.4.6 Sicurezza

Per sicurezza in un prodotto$_G$ software si intende il grado di protezione che il software ha sui dati per evitare che persone o servizi non autorizzati li ricevano.

- **Confidenzialità**: la capacità del software di assicurare la confidenzialità dei dati alle persone che hanno l'autorizzazione di accedere a tali informazioni;
- **Integrità**: la capacità del software di evitare modifiche o accessi non autorizzati;
- **Non ripudio**: la capacità del software di dimostrare che eventi o azioni sono avvenuti, in modo da non ripudiarli in un secondo momento;
- **Responsabilità**: la capacità del software di risalire a una sola entità a partire da una azione di una entità;
- **Autenticità**: la capacità del software di provare l'identità delle risorse utilizzate.

A.1.4.7 Manutenibilità

Per manutenibilità si intende la capacità di un prodotto$_G$ software di essere modificato e aggiornato senza l'introduzione di problemi e anomalie in seguito a cambiamenti dell'ambiente e/o dei requisiti.

- **Modularità**: la capacità del software di essere composto da componenti in modo che un cambio di un componente non abbia un grande impatto sul resto del sistema;
- **Riutilizzabilità**: la capacità del software di riutilizzare risorse più di una volta;
- **Analizzabilità**: la capacità del software di essere facilmente analizzabile per individuare un errore, le causa di un fallimento o di identificare le parti da modificare;
- **Modificabilità**: la capacità del software di essere facilmente modificabile senza introdurre errori non gestibili dal programmatore;
- **Testabilità**: la capacità del software di essere testato facilmente così da verificare ogni aggiunta e/o modifica.

A.1.4.8 Portabilità

Per portabilità si intende la caratteristica del software di essere utilizzato in modo efficacie quando esso viene trasferito su diversi hardware o software.

- **Adattabilità**: la capacità del software di essere adattato per hardware o software in continuo cambiamento;
- **Installabilità**: la capacità del software di essere installato o disinstallato efficacemente in un ambiete specifico;
- **Sostituibilità**: la capacità del software di rimpiazzare un altro prodotto$_G$ con lo stesso scopo nello stesso ambiente.