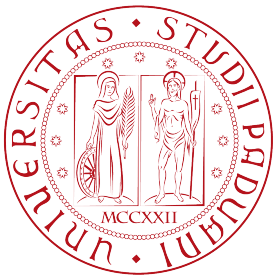
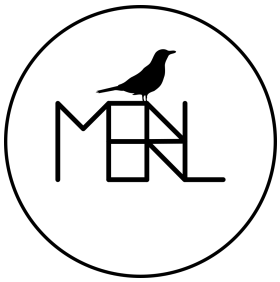


Università degli Studi di Padova
Laurea: Informatica
Corso: Ingegneria del Software
Anno Accademico: 2021/2022



Gruppo: MERL
Email: merlunipd@gmail.com

Norme di Progetto

Informazioni sul documento

Versione	V1.0.4
Uso	Interno
Data approvazione	08/03/2022
Distribuzione	Prof. *Vardanega Tullio*
	Prof. *Cardin Riccardo*
	Gruppo *MERL*

Registro delle Modifiche

Versione	Data	Autore	Verificatore	Modifica
v1.0.4	20/04/2022	Emanuele Pase	Marco Mamprin	Modificata sezione "Metriche"
v1.0.3	07/04/2022	Lorenzo Onelia	Riccardo Contin	Rimozione processi obsoleti nel capitolo "Processi Organizzativi"
v1.0.2	04/04/2022	Mattia Zanellato	Riccardo Contin	Aggiunta sottosezione "Metodo di lavoro"
v1.0.1	24/03/2022	Lorenzo Onelia	Emanuele Pase	Fix Minori
v1.0.0	08/03/2022	Marko Vukovic	-	Approvazione
v0.1.8	04/03/2022	Lorenzo Onelia	Mattia Zanellato	Aggiunta Lista di distribuzione
v0.1.7	21/02/2022	Mattia Zanellato	Emanuele Pase	Fix finali
v0.1.6	04/02/2022	Marco Mamprin	Mattia Zanellato	Aggiunta sezione "Miglioramento"
v0.1.5	02/02/2022	Marko Vukovic	Marco Mamprin	Aggiunta sezione "Glossario". Fix: "Processi Primari" e "Processi di Supporto". Fix minori
v0.1.4	15/01/2022	Marco Mamprin	Mattia Zanellato	Aggiunta sottosezione "Testing"
v0.1.3	09/01/2022	Marco Mamprin	Emanuele Pase	Aggiunta sottosezione "Metriche"
v0.1.2	07/01/2022	Riccardo Contin	Marko Vukovic	Aggiunte sottosezioni "Preventivo" e "Consuntivo"
v0.1.1	02/01/2022	Marko Vukovic	Emanuele Pase	Aggiunta sezione "Automazione"
v0.1.0	27/12/2021	Riccardo Contin	-	Approvazione

v0.0.13	24/12/2021	Emanuele Pase	Marco Mamprin	Fix minori
v0.0.12	24/12/2021	Lorenzo Onelia	Emanuele Pase	Fix minori
v0.0.11	21/12/2021	Marko Vukovic	Marco Mazzucato	Aggiunta sezione ”Gestione di Processo”
v0.0.10	18/12/2021	Emanuele Pase	Marco Mazzucato	Aggiunta sezione ”Sviluppo”
v0.0.9	17/12/2021	Mattia Zanellato	Lorenzo Onelia	Aggiunta sezione ”Fornitura”
v0.0.8	17/12/2021	Mattia Zanellato	Lorenzo Onelia	Aggiunto capitolo ”Introduzione”
v0.0.7	16/12/2021	Marco Mazzucato	Marco Mamprin	Aggiunta sezione ”Documentazione”
v0.0.6	14/12/2021	Marco Mamprin	Riccardo Contin	Aggiunta sezione ”Formazione”
v0.0.5	14/12/2021	Marco Mamprin	Riccardo Contin	Aggiunta sezione ”Validazione”
v0.0.4	14/12/2021	Marco Mamprin	Riccardo Contin	Aggiunta sezione ”Gestione della Qualità”
v0.0.3	11/12/2021	Lorenzo Onelia	Emanuele Pase	Aggiunta sezione ”Verifica”
v0.0.2	10/12/2021	Emanuele Pase	Riccardo Contin	Aggiunta sezione ”Gestione della Configurazione”
v0.0.1	08/12/2021	Marko Vukovic	Marco Mazzucato	Aggiunta sezione ”Infrastruttura”
v0.0.0	07/12/2021	Marko Vukovic	Marco Mazzucato	Creata prima struttura del documento

Indice

1. Introduzione

1.1 Scopo del Documento

Lo scopo del documento è quello di definire le regole che ogni membro del gruppo *MERL* deve rispettare al fine di raggiungere in modo efficace ed efficiente la creazione del prodotto finale.

Vengono inoltre specificate le convenzioni$_G$ sull'uso dei vari strumenti scelti per la realizzazione del prodotto e vengono illustrati i processi che saranno adottati dal gruppo.

1.2 Scopo del Prodotto

Al giorno d'oggi ogni servizio presente sul web richiede un'autenticazione$_G$ tramite login$_G$, fase fondamentale per la protezione dei dati di un individuo. Risulta ancora più importante se viene considerata la possibile presenza di malintenzionati con lo scopo di rubare ciò che dovrebbe essere privato. La presenza di attacchi informatici negli anni è andata aumentando e continua tuttora a crescere, per questo è necessario che questa pratica venga il più possibile riconosciuta e arginata.

Il capitolato C5 ha proprio come obiettivo quello di trovare una soluzione a questo problema. L'idea è quella di riconoscere le attività lecite e quelle illecite attraverso la raccolta, l'analisi e la visualizzazione di dati sotto forma di grafici e modelli che permettano un riconoscimento immediato delle differenze nei tentativi di accesso.

Con questo scopo il gruppo *MERL* si impegnerà nella realizzazione di un'applicazione web$_G$ in grado di leggere grandi quantità di dati di login per poi mostrare tramite dei grafici la natura di questi, riuscendo nell'intento di riconoscere a primo impatto le attività sospette.

1.3 Glossario

Al fine di evitare incomprensioni relative alla terminologia usata all'interno del documento, viene fornito un Glossario nel file *Glossario V1.0.0* in grado di dare una definizione precisa per ogni vocabolo potenzialmente ambiguo. Tali termini verranno evidenziati all'interno del documento con una G in pedice.

1.4 Riferimenti

1.4.1 Riferimenti normativi

- Capitolato d'appalto C5 - *Zucchetti S.p.A.*: *Login Warrior* https://www.math.unipd.it/~tullio/IS-1/2021/Progetto/C5.pdf.

1.4.2 Riferimenti informativi

- Slide T3 - Corso di Ingegneria del Software - Processi di ciclo di vita https://www.math.unipd.it/~tullio/IS-1/2021/Dispense/T03.pdf;
- Slide FC1 - Corso di Ingegneria del Software - Amministrazione di progetto https://www.math.unipd.it/~tullio/IS-1/2020/Dispense/FC1.pdf.

2. Processi Primari

2.1 Fornitura

2.1.1 Descrizione

Nella seguente sezione sono riportate tutte le regole che ogni membro del gruppo *MERL* si impegna a rispettare per mantenere attivi e produttivi i rapporti con il proponente$_G$ *Zucchetti S.p.A.*.

2.1.2 Scopo

Il processo di fornitura ha lo scopo di risolvere ogni dubbio legato alla volontà del proponente, inseguendo così la totale comprensione delle richieste al fine di soddisfarle a pieno.

2.1.3 Aspettative

Durante l'intera realizzazione del progetto il gruppo *MERL* intende mantenere rapporti frequenti con l'azienda *Zucchetti S.p.A.* al fine di evitare qualsiasi tipo di incomprensione dovuta alla mancanza di dialogo. Riteniamo di fondamentale importanza avere un riferimento in grado di darci supporto durante l'intero svolgimento. Di seguito i punti fondamentali a cui daremo primaria importanza in questo rapporto:

- Determinare i bisogni che il prodotto finale deve soddisfare;
- Stabilire le tempistiche di consegna del prodotto;
- Ottenere feedback$_G$ riguardo al lavoro svolto;
- Ottenere chiarimenti relativi a possibili dubbi o incomprensioni;
- Definire i vari vincoli sia riguardanti il prodotto finale sia riguardanti i processi intermedi che verranno attuati.

2.2 Sviluppo

2.2.1 Descrizione

Lo scopo del processo di sviluppo è quello di raggruppare compiti e attività relative alla codifica di un prodotto software, applicandole durante il suo intero ciclo di vita$_G$. Al fine di produrre un software che rispetti le aspettative del proponente è necessario:

- Determinare i vincoli tecnologici;
- Determinare gli obiettivi di sviluppo e design$_G$;
- Realizzare un prodotto software che superi tutti i test di verifica$_G$ e di validazione$_G$.

2.2.2 Attività

Il processo di sviluppo prevede l'esecuzione delle seguenti attività:

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

2.2.3 Analisi dei requisiti

L'analisi dei requisiti$_G$ è quell'attività che precede lo sviluppo e ha la funzione di:

- Definire lo scopo del prodotto da realizzare;
- Definire gli attori$_G$ del sistema;
- Fissare le funzionalità$_G$ del prodotto;
- Fornire una visione più chiara del problema ai progettisti;
- Fornire un riferimento ai verificatori per l'attività di controllo dei test;
- Fornire una stima della mole di lavoro.

Scopo

Lo scopo dell'attività è quello di fornire in un documento tutti i requisiti individuati. Al fine di individuarli è necessario:

- Leggere e comprendere la specifica del capitolato;
- Mantenere un confronto costante con il proponente.

Casi d'uso

Definiscono uno scenario in cui uno o più attori interagiscono con il sistema. Sono identificati nel modo seguente:

UC[Numero caso d'uso].[Sottocaso]-[Titolo caso d'uso]

Struttura dei requisiti

Il codice identificativo di ciascun requisito$_G$ è di seguito riportato:

R[Tipologia].[Importanza].[Codice]

con:

- Tipologia:
  - **F**: requisito funzionale (servizi e funzioni offerti dal sistema);
  - **Q**: requisito di qualità (vincoli di qualità, vedere il *Piano di Qualifica V1.0.0* per i dettagli);
  - **V**: requisito di vincolo (vincoli sui servizi offerti dal sistema);
  - **P**: requisito prestazionale (vincoli sulle prestazioni da soddisfare).
- Importanza:
  - **1** requisito obbligatorio;
  - **2** requisito desiderabile ma non obbligatorio;
  - **3** requisito opzionale.
- Codice:
  - Identificativo del requisito che risulta essere univoco in base alla tipologia. In alcuni casi il codice presenta dei sottocasi identificati a loro volta con un "." (punto) seguito dal corrispondente valore del sottocaso.

2.2.4 Progettazione

Scopo

Lo scopo della progettazione è quello di definire una possibile soluzione ai requisiti evidenziati dall'analisi.

Descrizione

La progettazione è formata da due parti:

- **Progettazione logica:** motiva le tecnologie, i framework$_G$ e le librerie$_G$ usate per la realizzazione di un prodotto, dimostrandone l'adeguatezza nel PoC$_G$. Contiene:

  - I framework e le tecnologie utilizzate;
  - Il Proof of Concept$_G$;
  - I diagrammi UML$_G$.

- **Progettazione di dettaglio:** illustra la base architetturale del prodotto coerentemente a ciò che è previsto nella Progettazione logica.
  Contiene:

  - Diagrammi delle classi;
  - Tracciamento delle classi;
  - Test di unità per ogni componente.

2.2.5 Codifica

Scopo

Lo scopo della codifica è quello di implementare le specifiche individuate in un prodotto utilizzabile.

Commenti

Nel caso sorga la necessità di scrivere qualche commento al codice è preferibile che esso sia chiaro e conciso.

Nomi dei file

I nomi dei file devono:

- Essere univoci;
- Esplicitare il contenuto dei file stessi.

3. Processi di Supporto

3.1 Documentazione

3.1.1 Scopo

Tutti i processi e le attività di sviluppo devono essere documentate al fine di poter tenere traccia, in modo più veloce e chiaro per tutti, di ciò che è stato fatto. La presente sezione ha l'obiettivo di annotare tutte le norme$_G$ che regoleranno il processo di documentazione durante l'intero ciclo di vita$_G$ del software, in modo che tutti i prodotti documentali risultino validi e coerenti dal punto di vista formale e tipografico.

3.1.2 Descrizione

Questa sezione contiene tutte le norme per la corretta stesura e verifica$_G$ dei documenti prodotti dai membri del gruppo. Ogni membro durante la redazione dei documenti è tenuto a seguire le regole presentate nella presente sezione.

3.1.3 Aspettative

Le aspettative di questo processo sono:

- Avere una struttura comune e chiara per i documenti nell'arco del ciclo di vita del software;
- Avere delle norme e convenzioni ben precise che coprono tutti gli aspetti della stesura di un documento, in modo da poter lavorare autonomamente.

3.1.4 Ciclo di vita di un documento

- **Pianificazione:** il documento viene pensato e vengono organizzate le varie parti. Questo accade soprattutto quando le informazioni sono numerose e complesse;
- **Impostazione:** viene creata la struttura con intestazione, header e footer. Il *Responsabile* quindi crea delle issue$_G$ su GitHub$_G$ che rappresentano ciascuna sezione del documento;

- **Realizzazione:** i membri assegnati a quel documento autonomamente scelgono una sezione tramite le issue di GitHub e i membri cominciano la stesura dei relativi contenuti;
- **Verifica:** ogni sezione viene verificata da un componente che non sia il redattore;
- **Approvazione:** terminata la verifica di tutte le sezioni il *Responsabile* rilegge il documento e decide se approvarlo o meno; se approvato viene pubblicato nel repository$_G$ altrimenti si torna alla fase di verifica.

3.1.5 Template in LaTeX

Il gruppo ha deciso di adottare il linguaggio LaTeX$_G$, grazie al quale viene standardizzata la struttura dei documenti. L'uso di un template$_G$ comune per la strutturazione dei documenti permette al gruppo di concentrarsi sulla stesura dei soli contenuti che verranno poi integrati facilmente grazie all'uso di questo linguaggio.

3.1.6 Documenti prodotti

I documenti prodotti saranno di due tipi:

- **Formali:** sono i documenti che regolano l'operato del gruppo e gli esiti delle attività dello stesso durante tutto il ciclo di vita del software. Le caratteristiche di questi documenti sono:
  - Storicizzazione delle versioni del documento prodotte durante la sua stesura;
  - Nomi univoci a ogni versione;
  - Approvazione della versione definitiva da parte del *Responsabile di Progetto*.

  Se un documento formale ha più versioni, si considera come valida sempre la più recente tra quelle approvate dal *Responsabile*.

  I documenti possono essere:
  - **Interni:** riguardano le dinamiche interne al gruppo di lavoro, poco utili a committente$_G$ e proponente$_G$;
  - **Esterni:** interessano committente e proponente e verranno loro consegnati nell'ultima versione approvata.

  Sono documenti formali:
  - ***Norme di Progetto*:** contiene le norme e le regole, stabilite dai membri del gruppo, alle quali ci si dovrà attenere durante l'intera durata del lavoro di progetto. Documento interno;

  - ***Glossario*:** elenco ordinato di tutti i termini usati nella documentazione che il gruppo ritiene necessitino di una definizione esplicita, al fine di rimuovere ogni possibile ambiguità. Documento esterno;
  - ***Piano di Progetto*:** espone la pianificazione di tutte le attività di progetto previste, presentando una previsione dell'impegno orario e un preventivo$_G$ delle spese. Documento esterno;
  - ***Piano di Qualifica*:** espone e descrive i criteri di valutazione della qualità adottati dal gruppo. Documento esterno;
  - ***Analisi dei Requisiti*:** espone tutti i requisiti e le caratteristiche che il prodotto finale dovrà avere. Documento esterno;
  - ***Manuale Utente*:** guida l'utente finale durante l'installazione e l'utilizzo delle funzionalità del prodotto. Documento esterno;
  - ***Specifica Architetturale*:** espone e descrive le scelte progettuali effettuate nella realizzazione del prodotto, attraverso diagrammi delle classi e di sequenza, scelta di design pattern e loro contestualizzazione al prodotto. Documento esterno.

- **Informali:** caratteristiche di questi documenti:
  - Non ancora approvati dal *Responsabile di Progetto*;
  - Non soggetto a versionamento.

  I documenti che appartengono alla seconda categoria sono i documenti presenti nel **Google Drive**$_G$ condiviso.

3.1.7 Struttura del documento

Ogni documento è formato da diverse sezioni, ognuna definita dal proprio file LaTeX. La parte principale è chiamata `"nomedoc.tex"` (dove `nomedoc` indica la sigla di quel documento, vedi sezione 3.1.16 Sigle) e contiene le seguenti componenti:

- La prima pagina che specifica il nome del documento;
- Il registro delle modifiche, che permette di tenere tracciabili i cambiamenti a quel documento;
- I file LaTeX delle sezioni con il contenuto vero e proprio del documento. Queste sezioni vengono decise da tutti i componenti del gruppo quando viene creato il documento.

3.1.8 Prima pagina

La prima pagina di un documento è formata da:

- Logo dell'Università di Padova con le informazioni del corso;
- Logo del gruppo con relativa email;
- Nome del documento.

3.1.9 Registro delle modifiche

Ogni documento presenta un registro delle modifiche sotto forma di tabella che tiene traccia dei cambiamenti significativi dello stesso durante il suo ciclo di vita. Ogni voce della tabella riporta:

- La versione del documento dopo la modifica;
- La data in cui è stata apportata tale modifica;
- Il nome dell'autore e del *Verificatore* della modifica;
- Una sintetica descrizione della modifica apportata.

3.1.10 Indice

Ogni documento presenta un indice dei contenuti, subito dopo il registro delle modifiche. Dove necessario, sono presenti anche un indice delle illustrazioni e uno delle tabelle presenti nel documento.

3.1.11 Verbali

I *Verbali* contengono la prima pagina come gli altri documenti ma non viene messo l'indice data la brevità di questi documenti. Il contenuto di un Verbale è così organizzato:

- **Informazioni generali**
  - Tipo di riunione, può essere interna o esterna;
  - Luogo;
  - Data;
  - Ora di inizio;
  - Ora di fine;
  - Moderatore;
  - Scriba;
  - *Verificatore*;
  - Partecipanti.
- **Diario della riunione**
  Riporta gli argomenti trattati durante la riunione nella seguente forma:
  - Elenco puntato per i *Verbali Interni*;
  - Elenco puntato e schema domanda-risposta per i *Verbali Esterni*.
- **Todo**
  Tabella che riassume le cose da fare (task) nell'immediato periodo. Viene anche specificato a chi è assegnata una determinata task.

3.1.12 Nome dei file

Il nome dei file e delle cartelle specifici di un singolo documento si chiamano con la sigla di quel documento. Tutti i file LaTeX che contengono le sezioni di un documento vanno scritti in minuscolo e separati da `"_"` se presentano più di una parola. I *Verbali* invece vengono salvati usando la data in cui è stata effettuata la riunione nel formato `aaaa.mm.gg`.

3.1.13 Stile di testo

Gli stili di testo adottati nei documenti sono:

- **Grassetto:** questo stile viene applicato alle parole ritenute particolarmente importanti;
- **Corsivo:** questo stile viene applicato ai nomi propri, ai nomi dei ruoli, al nome del progetto e ai nomi dei documenti;
- **Monospace:** questo stile viene applicato a snippet di codice;
- **Sottolineato:** questo stile viene applicato a link interni al documento.

3.1.14 Norme tipografiche

- Gli elenchi puntati e gli elenchi numerati iniziano sempre con la lettera maiuscola;
- Gli elementi di un elenco finiscono sempre con `";"` tranne l'ultimo che finisce con `"."`;
- La parola preceduta da `":"` deve sempre avere la prima lettera minuscola;
- Il nome dei ruoli sempre scritti per esteso (in corsivo) e con la lettera maiuscola (es: *Responsabile di Progetto*);
- Il nome dei documenti scritti con la maiuscola e per esteso a meno di ripetizioni (in tal caso si utilizza la sigla, e.g "*NdP*");
- Il nome dei capitoli (`chapter`) e delle sezioni (`section`) sempre con tutte le iniziali maiuscole a parte gli articoli e le preposizioni;
- Il nome delle sottosezioni (`subsection`) e inferiori solo la prima lettera maiuscola.

3.1.15 Glossario

Il *Glossario* è un documento contenente vocaboli tecnici o ambigui per cui il gruppo sente la necessità di definire una descrizione comune per poter comunicare in modo più conciso ed efficiente.

Il *Glossario* è un documento condiviso su Google Drive a cui hanno accesso in lettura e scrittura tutti i membri del gruppo. Se durante il progetto didattico si incontra qualche termine particolarmente significativo, qualsiasi membro del gruppo può inserirlo insieme alla relativa definizione all'interno del *Glossario*, verificando che esso non sia già presente.

Ciascun termine presente nel *Glossario* verrà segnalato all'interno di ciascun documento almeno una volta, tipicamente la prima volta che si incontra, con una G in pedice. Si evitano titoli e intestazioni di tabelle.

3.1.16 Sigle

- Sigle relative ai nomi dei documenti:
  - ***Analisi dei Requisiti:*** *AdR*;
  - ***Piano di Progetto:*** *PdP*;
  - ***Piano di Qualifica:*** *PdQ*;
  - ***Norme di Progetto:*** *NdP*;
  - ***Manuale Utente:*** *MU*;
  - ***Glossario:*** *G*;
  - ***Specifica Architetturale:*** *SA*;
  - ***Verbali Interni:*** *VI*;
  - ***Verbali Esterni:*** *VE*.
- Sigle relative alle revisioni:
  - ***Requirements and Technology Baseline:*** *RTB*;
  - ***Product Baseline:*** *PB*;
  - ***Customer Acceptance:*** *CA*.

3.1.17 Tabelle

Ogni tabella è contrassegnata da una didascalia descrittiva del contenuto, posta sotto di essa centrata rispetto alla pagina. Nella didascalia di ogni tabella viene indicato l'identificativo

Tabella [X]: Descrizione

dove [**X**] indica il numero assoluto della tabella all'interno del documento e **Descrizione** contiene una breve descrizione di ciò che può essere visualizzato nella tabella. Le tabelle delle modifiche invece non seguono questa regola perché non hanno didascalia.

3.1.18 Immagini

Le immagini sono contenute nella cartella chiamata `immagini` che a sua volta è contenuta nella cartella del documento a cui si riferiscono. Verranno poi richiamate dentro i file LaTeX ove servono. Per esempio i diagrammi UML_G vengono riportati come immagini.

3.1.19 Preventivo

Questa è una sezione molto importante del *Piano di Progetto*. Per ogni periodo che si pianifica si vuole indicare:

- I ruoli che opereranno;
- Preventivo orario:
  - Tabella con una riga per membro e una colonna per ruolo che rappresenta la distribuzione delle ore;
  - Istogramma_G che rappresenta graficamente ciò che esprime la tabella precedente;
  - Un grafico_G a torta che presenta le percentuali di lavoro per ogni ruolo.
- Preventivo economico:
  - Tabella con una riga per ruolo, una colonna con le ore per ruolo e il costo complessivo di ogni ruolo;
  - Un grafico a torta che presenta le percentuali di costo per ogni ruolo.

3.1.20 Consuntivo

Questa è una sezione molto importante del *Piano di Progetto*. Per ogni periodo pianificato e per cui si è fatto un preventivo si vuole indicare:

- Consuntivo_G orario:
  - Tabella con la struttura identica a quella del preventivo orario dove si indicano le ore consuntivate e tra parentesi tonde la differenza tra le ore consuntivate e quelle preventivate;
  - Tabella con la struttura identica a quella del preventivo orario dove si indicano le ore rimaste per ruolo e per persona.
- Consuntivo economico:
  - Tabella con la struttura identica a quella del preventivo economico dove si indicano in una colonna le ore consuntivate e tra parentesi tonde la differenza tra ore consuntivate e preventivate, e nell'altra colonna i costi consuntivati con la corrispondente differenza, sempre tra parentesi tonde, tra i costi consuntivati e quelli preventivati.

- Conclusioni: alla luce delle differenza tra preventivo e consuntivo si analizza quelli che possono essere i problemi o i punti di forza del periodo appena concluso.

3.2 Gestione della Configurazione

3.2.1 Scopo

Lo scopo di questa sezione è quello di espilicitare come il gruppo intende gestire la produzione di risorse durante l'intero sviluppo del progetto.

3.2.2 Aspettative

Le attese che il gruppo *MERL* si pone sono:

- Semplificare l'individuazione di conflitti ed errori;
- Uniformare gli strumenti utilizzati;
- Tracciare le modifiche e avere sempre a disposizione una versione precedente per ogni file.

3.2.3 Descrizione

Il processo di gestione della configurazione$_G$ ha lo scopo di creare un ambiente in cui la produzione di documentazione e di codice avviene in maniera sistematica, ordinata e standardizzata. Ciò è permesso raggruppando ed organizzando tutte le regole e gli strumenti adoperati.

3.2.4 Versionamento

Codice di versione

La storia di una risorsa deve sempre poter essere ricostruibile in quanto nel suo arco di vita essa subisce svariate modifiche. A tal scopo è fondamentale l'introduzione di un sistema di identificazione della versione:

[X].[Y].[Z]

Dove:

- **X** Versione di produzione;
- **Y** Integrazione di piccoli incrementi;
- **Z** Incremento di piccole dimensioni verificato dai verificatori e dal *Responsabile*.

3.2.5 Sistemi software utilizzati

Per la gestione delle differenti versioni è stato deciso di utilizzare il sistema di versionamento distribuito Git$_G$; utilizzando il servizio GitHub per gestire il repository. Per la suddivisione e la gestione dei lavori da svolgere il nostro gruppo utilizza il sistema di Issue e Pull Request$_G$ di GitHub.

3.2.6 Struttura repository

Repository utilizzata

Il repository in cui tutti i documeti vengono caricati è pubblico e si trova al seguente indirizzo: https://github.com/merlunipd/login-warrior.

Organizzazione del repository

L'organizzazione del repository è ispirata a **GitHub Flow** ed è riassunta di seguito:

- Ramo principale **Main** in cui è presente la documentazione, o le parti dei documenti, approvate da almeno un *Verificatore* e dal *Responsabile*;
- Rami secondari utilizzati per redarre i vari paragrafi della documentazione che, una volta uniti al ramo principale attraverso l'attività di merging, vengono eliminati per evitare di ostruire il repository.

Nel ramo Main i documenti si trovano all'interno della cartella "documenti", all'interno della quale possiamo trovare:

- La cartella "candidatura" che contiene i documenti inerenti alla candidatura al capitolato C5;
- La cartella "interni" che contiene i documenti utili al gruppo;
- La cartella "esterni" contenente i documenti da fornire ai committenti e al proponente.

Sempre nel Main possiamo trovare il file ".gitignore" necessario ad evitare il tracciamento dei file LaTeX non necessari ai fini del progetto.

3.3 Gestione della Qualità

3.3.1 Scopo

L'obiettivo della gestione della qualità è quello di garantire che i processi e il prodotto soddisfino le richieste del proponente e che lo facciano con la miglior qualità possibile. Vogliamo, inoltre, poter ottenere un miglioramento continuo della nostra qualità, osservando il nostro andamento e tramite verifiche retrospettive$_G$.

3.3.2 Piano di Qualifica

Per coprire gli obiettivi di questo processo utilizziamo il *Piano Di Qualifica V1.0.0*, cioè un documento nel quale:

- Fissiamo gli obiettivi di qualità;
- Definiamo le metriche per avere una visione quantitativa;
- Definiamo dei test di qualità e funzionamento;
- Effettuiamo e documentiamo i test;
- Visualizziamo un cruscotto$_G$ dello stato attuale;
- Discutiamo dei vari incontri con una verifica delle retrospettive;
- Proponiamo idee di auto-miglioramento.

3.3.3 Testing

All'interno del *Piano di Qualifica V1.0.0* troviamo gli obiettivi di qualità del processo e del prodotto, con le relative metriche per verificarne l'accettabilità, successivamente abbiamo dei test$_G$ che garantiscono la qualità generale del nostro software, che appartengono alle seguenti categorie:

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

Per questi ultimi test verrà indicato se sono stati implementati o meno.
Infine troviamo un resoconto che contiene i risultati dei test della qualità di processo, della qualità di prodotto e del software, in cui, grazie a opportuni grafici, abbiamo una rappresentazione dei vari indici nel corso del tempo. Per svolgere al meglio questo lavoro è possibile affidarsi alla sottosezione 3.3.4 Metriche, in cui è presente la descrizione e le formule delle varie metriche.

3.3.4 Metriche

All'interno del *Piano di Qualifica V1.0.0* vengono utilizzate diverse metriche, di seguito viene spiegato come capirle e come utilizzarle per eseguire i test di qualità.

Codifica

M[Tipologia][Id numerico]

Dove:

- **Tipologia** indica il tipo della metrica$_G$ e può essere:
  - **PC** per processo;
  - **PD** per prodotto.
- **Id numerico** indica un numero univoco incrementale separato per le due tipologie.

Descrizione

Troviamo di seguito per ogni obiettivo di qualità la descrizione delle sue metriche:

- **Comprensione**,
  - E_o, numero errori ortografici;
  - N_p, numero di parole;
  - N_f, numero di frasi;
  - N_l, numero di lettere.

Metrica	Nome	Descrizione	Formula
MPD1	Errori Ortografici	Percentuale di errori ortografici e grammaticali in un documento	$\frac{E_o}{N_p} * 100$
MPD2	Indice di Gulpease	Indice di leggibilità di un testo in base all'istruzione ottenuta	$89 + \frac{300 * N_f - 10 * N_l}{N_p}$

Per facilitare il calcolo dell'indice di Gulpease$_G$ è possibile affidarsi al seguente calcolatore online:
https://www.andreapacchiarotti.it/archivio/gulpease-indice.html

- **Funzionalità**,
  - R_s, numero requisiti soddisfatti;
  - R_t, numero requisiti totali.

Metrica	Nome	Descrizione	Formula
MPD3	Copertura dei requisiti	Percentuale di requisiti trovati nell'*Analisi dei Requisiti V1.0.0* coperti dal software	$\frac{R_s}{R_t} * 100$

- **Efficienza**,

Metrica	Nome	Descrizione	Formula
MPD4	Tempo di risposta medio	Tempo medio impiegato dal software per rispondere ad un'azione dell'utente	-

- **Usabilità**,

Metrica	Nome	Descrizione	Formula
MPD5	Tempo apprendimento	Tempo impiegato dall'utente per imparare ad usare il software	-
MPD6	Raggiunta dell'obiettivo	Numero di click impiegato dall'utente per raggiungere il suo scopo	-
MPD7	Errori dell'utente	Numero di errori fatti dall'utente per raggiungere il suo scopo	-

- **Affidabilità**,
  - T_b, numero test andati a buon fine;
  - T_t, numero test totali;
  - E_b, numero test con errore andati a buon fine;
  - E_t, numero test con errore totali.

Metrica	Nome	Descrizione	Formula
MPD8	Maturità dei test	Test andati a buon fine su tutti i test eseguiti(per ogni obiettivo)	$\frac{T_b}{T_t} * 100$
MPD9	Gestione degli errori	Test eseguiti con errori andati comunque a buon fine	$\frac{E_b}{E_t} * 100$

- **Manutenibilità**,
  - R_c, numero righe di commenti;
  - R_t, numero righe di codice totali.

Metrica	Nome	Descrizione	Formula
MPD10	Comprensibilità del codice	Percentuale di righe di commenti sulle righe di codice totali	$\frac{R_c}{R_t} * 100$

- **Portabilità**,
  - O_s, numero OS supportati;
  - O_t, numero OS totali;
  - B_s, numero browser supportati;
  - B_t, numero browser totali.

Metrica	Nome	Descrizione	Formula
MPD11	OS supportati	Percentuale di sistemi operativi supportati dal software	$\frac{O_s}{O_t} * 100$
MPD12	Browser supportati	Percentuale di browser supportati dal software	$\frac{B_s}{B_t} * 100$

- **Controllo**,
  Alcuni valori importanti per la tabella successiva:
  - LC, indica la percentuale del lavoro completato;
  - LP, indica la percentuale del lavoro pianificato;
  - BAC (Budget at Completion), corrisponde al budget totale allocato inizialmente per il progetto;
  - AC (Actual Cost), indica i soldi spesi per il progetto fino al momento del calcolo;
  - ETC (Estimate to Complete), valore stimato delle attività necessarie per il completamento del progetto.

Metrica	Nome	Descrizione	Formula
EAC	Estimated at Completion	Indica il budget totale allocato per il progetto	AC + ETC
EV	Earned Value	Rappresenta il valore prodotto dal progetto fino al momento della misurazione in seguito alle attività svolte	EAC * LC
PV	Planned value	Corrisponde al denaro che si dovrebbe guadagnare al momento della misurazione	BAC * LP
SV	Schedule Variance	Descrive se il gruppo sta rispettando o meno i tempi prestabiliti per i processi	$(\frac{EV}{PT} - 1) * 100$
BV	Budget Variance	Descrive se il gruppo sta rispettando o meno i costi prestabiliti per i processi	$(\frac{EV}{AC} - 1) * 100$

- **Prodotto**,
  Alcuni valori importanti per la tabella successiva:
  - TT, numero di test totali;
  - PT, numero di test passati;
  - FT, numero di test falliti;
  - TQM (Total number of Quality Metrics), numero totale di metriche di qualità;
  - NQMS (Number of Quality Metrics Satisfied), numero di metriche di qualità soddisfatte;
  - Rc, righe totali di commenti;
  - Rt, righe codice totali;
  - LCE (Line of Code Executed), linee codice eseguite dai test;
  - LC (Line of Code), linee codice totali;
  - NoC (Number of Changed), numero requisiti cambiati;
  - NoD (Number of Deleted), numero requisiti eliminati;
  - NoA (Number of Added), numero requisiti aggiunti;
  - TNIR (Total Number of Initial Requirement), Numero totale requisiti iniziale;

Metrica	Nome	Descrizione	Formula
PTCP	Passed Test Cases Percentage	Indica il valore percentuale di test passati	$\frac{PT}{TT} * 100$
FTCP	Failed Test Cases Percentage	Indica il valore percentuale di test falliti	$\frac{FT}{TT} * 100$
QMS	Quality Metrics Satisfied	Indica la percentuale di metriche di qualità soddisfatte	$\frac{NQMS}{TQM} * 100$
CC	Code coverage	Descrive quanto il codice prodotto è coperto dalla suite di test dinamici	$\frac{LCE}{LC} * 100$
RSI	Requirements stability index	Indica quanto i requisiti variano nel tempo	$(1 - \frac{NoC*NoD*NoA}{TNIR}) * 100$

3.3.5 Aspettative

Con questo processo ci poniamo le seguenti aspettative:

- Assicurarci della qualità del prodotto che realizziamo;
- Assicurarci delle qualità dei processi e del gruppo;
- Poter avere una visione quantitativa del nostro avanzamento;
- Poter effettuare test frequentemente e che siano predicibili;
- Poterci migliorare progressivamente per non ripetere gli stessi errori;
- Soddisfare al meglio le aspettative del committente.

3.4 Automazione

3.4.1 Descrizione

Uno degli strumenti più potenti a disposizione di un informatico è l'automazione. Applicabile a operazioni ripetitive, tediose, sensibili, semplici o complesse, l'automazione delle operazioni può portare grandi vantaggi, tra cui:

- Riduzione delle risorse utilizzate (in particolare, nel nostro caso, tempo);
- Riduzione di errori umani.

I costi di automazione possono però essere molto elevati e non è possibile automatizzare tutti i processi. Risulta quindi importante selezionare con cura le attività da automatizzare o semi-automatizzare, in modo da mantenere ridotti i costi (in termini di ore investite) e ottenere dei benefici.

3.4.2 Scopo

Tenere traccia delle attività automatizzate e fornire documentazione sull'utilizzo degli strumenti.

3.4.3 Script

Lo "scripting"$_G$ è l'attività di scrittura di codice utilizzata al fine di automatizzare l'esecuzione di attività. Rispetto all'utilizzo di strumenti preesistenti ha il vantaggio di essere estremamente più flessibile. Inoltre per un gruppo di studenti di informatica, il costo iniziale di apprendimento è relativamente basso per attività semplici e chiare.

Il gruppo non impone particolari limitazioni sui linguaggi utilizzabili, ricordando però che:

- Il codice è soggetto a versionamento ed è posto nel repository;
- Il codice scritto va adeguatamente documentato, sia con annotazioni nello script (dettagli implementativi), sia in questo documento (descrizione ed utilizzo);
- Uno script ha dimensioni estremamente ridotte (tipicamente qualche decina di righe di codice, occasionalmente qualche centinaio).

`update_documents`

- **Descrizione**: script per compilare i documenti LaTeX in PDF e aggiornare il sito che li espone;
- **Utilizzo**:
  - **Collettivo**: utilizzato per aggiornare il repository condiviso, tipicamente utilizzato nel seguente modo:

```
# Riallineamento del repository locale e remoto
git checkout main
git pull

# Esecuzione dello script, supponendo di
# trovarsi nella directory radice del repository
python ./automazione/script/update_documents/main.py
# Se non ci sono errori durante l'esecuzione, procedere

# Aggiornamento del repository locale e remoto
git add -A
```

```
git commit -m "Aggiornamento dei documenti PDF e sito"
git push origin main
```

NB 1: non andando a modificare file sorgenti, il seguente commit$_G$ è accettato nel repository locale senza bisogno di pull request.

NB 2: tipicamente sarà compito del *Responsabile di Progetto* tenere il sito e i documenti esposti sufficientemente aggiornati;

– **Personale**: se un membro del gruppo desidera avere una versione di tutti i documenti più aggiornati rispetto a quelli esposti dal repository condiviso, può creare un **branch locale ad utilizzo personale** dove tenere documenti e sito aggiornati. Tale operazione può essere effettuate come segue:

```
# Checkout al branch locale personale
git checkout <nome_branch_locale_personale>
# Aggiornamento del branch rispetto al main
git merge main

# Esecuzione dello script, supponendo di
# trovarsi nella directory radice del repository
python ./automazione/script/update_documents/main.py
# Se non ci sono errori durante l'esecuzione, procedere

# Aggiornamento del repository locale (non remoto)
git add -A
git commit -m "Aggiornamento locale dei documenti PDF e sito"
```

3.5 Verifica

3.5.1 Scopo

La verifica ha come obiettivo il controllo che un prodotto sia corretto e completo.

3.5.2 Descrizione

Nella verifica si prende in input ogni processo e lo si controlla, in modo da identificare dubbi o incorrettezze. Si applica la verifica su un processo quando:

- Si raggiunge un livello di maturità adeguato e sufficiente;
- A seguire di un cambiamento di stato.

3.5.3 Indicazioni

Per assicurare il corretto svolgimento della verifica si devono rispettare i seguenti punti:

- Seguire procedure definite;

- Seguire criteri validi e affidabili;
- Ogni prodotto deve passare attraverso fasi successive, che verranno quindi verificate.

Una volta terminata la fase di verifica, rispettando i punti appena citati si può quindi proseguire alla fase di **Validazione**$_G$.

3.5.4 Verifica della documentazione

Il processo di verifica per quanto riguarda la documentazione può essere riassunto in queste attività:

- Controllo dell'ortografia e della sintassi;
- Controllo dell'aderenza alle convenzioni tipografiche e di stile adottate;
- Controllo della pertinenza e della correttezza dei contenuti del documento.

Attività di analisi statica

Questa tipologia di analisi risulta molto utile per la verifica di un documento. Si divide in:

- **Walkthrough**$_G$: tecnica dove il *Verificatore* va alla ricerca di eventuali errori attraverso una lettura ad ampio spettro;
- **Inspection**$_G$: tecnica dove il *Verificatore* va alla ricerca di specifici errori attraverso letture mirate.

Inizialmente l'attività di Walkthrough sarà prevalente rispetto a Inspection ma con l'avanzamento dell'attività di progetto e quindi con il ripetersi del processo di Verifica sulla documentazione, si riuscirà a sviluppare una lista degli errori più comuni detta **Lista di Controllo** consentendo l'utilizzo più massivo della tecnica **Inspection** che risulta essere più efficiente.

3.5.5 Verifica del prodotto

Al fine di minimizzare gli errori il codice viene controllato mediante attività di analisi statica e dinamica.

Attività di analisi statica

Lo strumento utilizzato per controllare in maniera statica il codice è *ESLint*, il quale permette di verificare che i principi di buona programmazione decisi vengano rispettati.

Attività di analisi dinamica

Per offrire una maggior garanzia che il prodotto non incorra in problemi durante l'esecuzione, il gruppo utilizza un insieme di test eseguibili a run-time. L'esecuzione di tali test deve essere ripetibile, perciò è necessario identificare vari strumenti per renderla automatica. A questo fine il gruppo *MERL* si affida al software *Jest.js*.

3.6 Validazione

3.6.1 Descrizione

Con Validazione si intende l'incontro con il committente in cui si presenta il proprio lavoro, in particolare il prodotto finale, ottenendo così l'approvazione o meno sull'operato.
Lo scopo è quello di accertarsi di aver conseguito i requisiti obbligatori minimi, imposti dal committente e in modo efficace ed efficiente. Per verificare il tutto si eseguirà un collaudo con il committente, che sarà composto da diversi test, che ne verifichino le qualità, e inoltre dovranno già essere eseguiti in precedenza ed essere predicibili, cioè senza risultati o comportamenti inattesi.

4. Processi Organizzativi

4.1 Gestione di Processo

4.1.1 Coordinamento

Comunicazioni

Le **comunicazioni interne** sono principalmente tra studenti del gruppo e sono conversazioni tra pari. Dipendentemente dal canale di comunicazione scelto, per favorire la produttività, bisognerà adottare un registro adeguato e seguire le convenzioni decise (consultare la sezione 4.2 Infrastruttura per più informazioni sugli strumenti di organizzazione utilizzati):

- **GitHub**$_G$: comunicazioni formali, brevi e concise di natura principalmente tecnica;
- **Discord**$_G$: in base alla categoria del canale
  - **Risorse**: comunicazioni formali di varia lunghezza che permettono al team di essere sempre aggiornato;
  - **Testuali**: comunicazioni di vario tipo: formali (ad esempio nel canale `generale`), informali (ad esempio nel canale `off-topic`);
  - **Vocali**: comunicazioni di vario tipo in accordo con la situazione (ad esempio: semi-formali per riunioni interne; informali per "chiacchiere tra colleghi").
- **WhatsApp**$_G$: in base alla chat utilizzata
  - **Gruppo**: comunicazioni semi-formali, brevi e strettamente inerenti al progetto;
  - **Individuale**: comunicazioni informali.
- **Google Calendar**: comunicazioni formali, brevi e concise in forma di informazioni complementari ad un evento di calendario.

Le **comunicazioni esterne** sono principalmente con committente e proponente. Queste hanno generalmente un valore molto più elevato e vanno adeguatamente preparate. Gli strumenti principali coinvolti sono: **Zoom**$_G$ e **Google Mail** (attraverso l'indirizzo di gruppo `merlunipd@gmail.com`). Il registro è sempre formale e dipendentemente dal contesto verrà utilizzato un vocabolario tecnico adeguato. È compito del *Responsabile di Progetto* gestire le comunicazioni esterne.

Riunioni

Le **riunioni interne** sono principalmente tra membri del gruppo. Per come è attualmente organizzato il gruppo vengono svolte a necessità, tipicamente non passano più di 10 giorni tra una e l'altra. È compito del *Responsabile di Progetto* organizzare tali incontri in data e orario che permetta all'intero gruppo di partecipare. In caso di problemi, si prova ad accordarsi in modo da poter partecipare tutti. Nell'eventualità ciò non fosse possibile, chi non riesce a partecipare potrà avere accesso al verbale dell'incontro e se necessario il *Responsabile* si occuperà di fornire tutti gli aggiornamenti fondamentali. I *Verbali* vengono redatti con modalità round-robin dai membri del gruppo.

Per mantenere efficiente il tempo di lavoro sincrono delle riunioni interne, si adottano i seguenti accorgimenti:

- Scaletta: ciascuna riunione segue una scaletta standard (riportata sotto);
- Moderazione: lo scriba del verbale avrà anche il ruolo di moderatore, cercando di mantenere la durata della riunione pari o inferiore a quanto stabilito ma riuscendo a discutere di tutti gli argomenti pianificati in modo completo;
- Preparazione: ciascun membro del gruppo si impegna a partecipare in modo attivo e produttivo agli incontri e questo richiede un minimo di preparazione pregressa. Il tempo di lavoro sincrono è prezioso e va utilizzato per confrontarsi su argomenti di cui si ha un certo grado di competenza (quantomeno avere un'idea di "cosa so" e "cosa non so").

La scaletta di una tipica riunione interna è la seguente:

- Controllo task boards: verifica e possibile discussione di tasks (GitHub) che necessitano di essere discusse tutti insieme;
- Attività specifiche di riunione: attività pianificate ad hoc per una particolare riunione;
- Tempo di slack: utilizzato per discutere di argomenti non previsti. Se l'attività di pianificazione è efficace e non ci sono imprevisti, questo spazio non sarà utilizzato.

È compito del *Responsabile di Progetto* riportare, espandere con le attività ad hoc e tenere aggiornata la scaletta di ciascun incontro sul calendario Google Calendar condiviso.

Le **riunioni esterne** sono principalmente quelle con committente e proponente. Generalmente saranno richieste dal gruppo in caso di necessità di opinioni più esperte su questioni tecniche o di way of working$_G$ e per controllare il corretto progresso del progetto didattico. Il tempo richiesto ad un esperto ha un altissimo valore, quindi le riunioni esterne dovranno avere un adeguato livello di preparazione e avere una durata contenuta.

Reperibilità

Ogni membro del gruppo è libero di organizzare il proprio tempo di lavoro individuale asincrono come preferisce, in accordo con altri impegni accademici, personali e quanto dichiarato nel preventivo di periodo.

Come compromesso tra efficacia di comunicazione asincrona e protezione delle tempo personale, i membri del gruppo si impegnano ad essere reperibili per questioni relative al progetto didattico col seguente orario: **dal lunedì al venerdì dalle 9 alle 17**. Qualsiasi estensione o riduzione di orari o giorni di reperibilità può essere concordata con i membri del gruppo. Questo orario di reperibilità non va considerato né utilizzato come tempo di lavoro attivo ma rappresenta un limite di disponibilità da offrire ai compagni di progetto in caso di necessità.

4.1.2 Pianificazione

Ruoli di progetto

I ruoli che ciascun membro dovrà ricoprire durante il corso del progetto, per un tempo congruo rispetto a quanto preventivato, sono:

- ***Responsabile di Progetto***: ha il ruolo di guidare il progetto a livello macroscopico e gestire lo svolgimento dei processi, in particolare:
  - Essere sempre aggiornato sullo stato di progresso del progetto;
  - Gestire la pianificazione delle attività di ciascuna milestone$_G$, definendole insieme a chi segue il documento *Piano di Progetto* e predisponendo le relative task su GitHub;
  - Approvare qualsiasi task sia stata completata e verificata, quindi processi primari e di supporto (su GitHub, approvando le pull request$_G$);
  - Gestire il calendario condiviso;
  - Gestire le comunicazioni esterne.
- ***Amministratore***: ha il ruolo di garantire l'efficacia$_G$ e l'efficienza$_G$ dei processi, in particolare:
  - Redigere i documenti: *Norme di Progetto*, *Piano di Progetto*, *Piano di Qualifica*;
  - Gestire l'infrastruttura e gli strumenti utilizzati;
  - Automatizzare i processi;
  - Individuare punti di miglioramento nei processi.
- ***Analista***: ha un ruolo fondamentale nelle fasi iniziali del progetto, comprendendo a fondo le necessità del proponente ed individuando i requisiti fondamentali che la fase di progettazione dovrà soddisfare. Nello specifico si occupa di:
  - Redigere il documento: *Analisi dei Requisiti*;

  - Studiare il dominio applicativo relativo alle richieste del proponente;
  - Scomporre le esigenze del proponente in elementi atomici da poter risolvere singolarmente.

- ***Progettista***: ha il compito di modellare i requisiti individuati nella fase di analisi e ricomporli in un'architettura$_G$ che possa soddisfarli. Nello specifico si occupa di:
  - Produrre un'architettura che soddisfi i requisiti richiesti;
  - Approfondire conoscenze tecniche e ricercare strumenti tecnologici utili nell'ambito di applicazione;
  - Produrre una soluzione con un alto livello di manutenibilità$_G$, seguendo le "best practices" note;
  - Produrre l'architettura di un sistema con il minimo numero di dipendenze possibili (basso grado di accoppiamento, alto grado di coesione tra componenti).

- ***Programmatore***: ha il ruolo di implementare l'architettura prodotta nella fase di progettazione, in particolare:
  - Scrivere codice che soddisfa le specifiche di progettazione;
  - Conoscere ed applicare le "best practices" note riguardanti la scrittura di codice;
  - Permettere un alto grado di manutenibilità del codice, versionando e documentando;
  - Scrivere i test relativi al codice prodotto;
  - Redigere il documento: *Manuale Utente*.

- ***Verificatore***: si occupa di controllare che le attività svolte rispettino il livello di qualità atteso. Per questioni ovvie, il *Verificatore* non può effettuare il controllo di un'attività svolta da lui stesso.

Il ruolo di *Responsabile* verrà assegnato a turno a tutti i membri del gruppo per una durata di circa **due settimane**. In questo modo sarà possibile per i membri del gruppo (possibilmente tutti) ricoprire questo ruolo in diversi stadi di avanzamento del progetto, beneficiando nei secondi mandati dell'esperienza maturata.

Gli altri ruoli di progetto non saranno assegnati a periodo ma ad attività. Quest'organizzazione più fluida ha lo scopo di aumentare la flessibilità del gruppo.

Gestione delle task

In generale il gruppo è organizzato per predisporre tutte le attività di progresso per ciascuna milestone, in modo che i membri del gruppo possano autonomamente autoassegnarsi le task (in assenza di accordi preesistenti) e lavorare il più possibile in parallelo e in asincrono. Il compito di definizione, predisposizione e configurazione$_G$ delle task viene svolto subito dopo la pianificazione delle milestone dal *Responsabile*

di Progetto seguendo quanto previsto dal *Piano di Progetto V1.0.0*. Per task di processi primari o di supporto (come la documentazione), si utilizza GitHub. Il loro ciclo di vita ha il seguente schema:

- Creazione: la task definita viene aperta sotto forma di "Issue"$_G$ su GitHub;
- Auto-assegnazione: un membro del gruppo può autonomamente prendere in carico una task non assegnata;
- Completamento: la task viene completata, tipicamente su un branch$_G$ distinto;
- Pull request$_G$: viene fatta una pull request del branch contenente l'attività completata, collegando la richiesta alla relativa issue;
- Verifica: un *Verificatore* effettua il controllo della qualità;
- Accettazione: quando il *Verificatore* conferma la validità della task, il *Responsabile di Progetto* esegue un rapido controllo e se tutto è in ordine approva la pull request, chiude la issue relativa e cancella il branch dell'attività.

In generale le dimensioni di una task sono variabili rispetto al processo considerato, in pratica sono attività non triviali ma sufficientemente contenute da poter essere svolte da un individuo in un tempo ragionevole. Ad esempio:

- (processo primario) Codifica di una classe;
- (processo di supporto) Stesura di una sezione di un documento;
- (processo organizzativo) Creazione e configurazione delle attività di una milestone.

Sia le task primarie e di supporto, sia quelle organizzative, hanno la possibilità di essere discusse individualmente utilizzando la sezione "Commenti" della rispettiva piattaforma. Questa funzionalità risulta fondamentale nel momento in cui l'autore e il *Verificatore* di una task hanno bisogno di instaurare una discussione. Questo approccio favorisce anche la tracciabilità delle attività svolte e dei relativi problemi incontrati.

Metodo di lavoro

In seguito alla prima revisione il gruppo, accortosi del ritardo accumulato, ha deciso di introdurre un nuovo metodo per rendere più efficace il lavoro svolto. A questo proposito è stato deciso, dopo varie considerazioni, di aggiungere la tecnica degli *Sprint*$_G$ del framework *Scrum*$_G$.
Questa tecnica permette di dividere il tempo di lavoro in brevi intervalli in modo da definire a priori le attività da svolgere con la possibilità di valutare ciò che è stato o non è stato fatto al termine dello sprint.
Ogni sprint, della durata di una settimana, prevede le seguenti fasi:

- **Sprint Planning**: pianificazione da svolgere in gruppo il primo giorno dello sprint. Questa fase prevede le seguenti attività:

  - Brainstorming delle idee: ogni membro esprime le proprie idee su ciò che ha più rilevanza nell'immediato futuro;
  - Obiettivi e issue: il *Responsabile* è incaricato di creare lo **Sprint Backlog** definendo gli obiettivi per lo sprint e di inserire le corrispondenti issue su *GitHub*;
  - Preventivi: ogni membro, in base alle attività che si impegna a svolgere durante lo sprint, indica il proprio preventivo orario.

- **Sprint Review**: revisione dello sprint appena trascorso da svolgere l'ultimo giorno dello sprint con la presenza di tutto il gruppo. Le attività che compongono questa fase sono le seguenti:

  - Consuntivo e produttività individuale: ogni membro esprime ciò che ha svolto, con successo o insuccesso, durante lo sprint. Se necessario può dimostrare in live i propri dubbi o risultati sull'attività svolta;
  - Obiettivi raggiunti: in seguito alle attività svolte dai membri viene stilata una lista di obiettivi raggiunti.
  - Obiettivi non raggiunti: in seguito alle attività non svolte dai membri viene stilata una lista di obiettivi non raggiunti. Questa lista andrà ad aggiungersi ai nuovi obiettivi per lo sprint successivo.

- **Sprint Retrospective**: retrospettiva con lo scopo di concludere definitivamente lo sprint appena svolto e di valutare l'andamento di quest'ultimo. È importante in questa fase valutare ciò che è andato bene, in modo da continuare su quella strada, e ciò che è andato male, per poter risolvere quelle particolari problematiche. Risulta utile stilare due liste:

  - Good: indica ciò che è andato bene durante lo sprint;
  - To Improve: indica ciò che può essere migliorato.

  In particolare questa attività permette di definire ciò che è necessario:

  - Iniziare a fare;
  - Smettere di fare;
  - Continuare a fare.

4.2 Infrastruttura

Fanno parte dell'infrastruttura organizzativa tutti gli strumenti che permettono al gruppo di attuare in modo efficace ed efficiente i processi organizzativi. In particolare tali strumenti permettono la **comunicazione**, il **coordinamento** e la **pianificazione**.

4.2.1 Strumenti

GitHub

È il principale servizio di hosting della repository$_G$ di gruppo e di controllo della versione distribuito.

Generalmente il workflow adottato dal gruppo è il GitHub Flow, che sinteticamente segue il seguente schema:

- Riallineamento della repository locale con quella remota;
- Creazione di un branch locale su cui effettuare le modifiche;
- Push del branch locale verso repository remota;
- Creazione di una pull request;
- Verifica e successivo merge$_G$ del branch con le modifiche;
- Eliminazione del branch utilizzato dalla repository remota.

Per i dettagli consultare la documentazione ufficiale:

- https://docs.github.com/en/get-started/quickstart/github-flow.

GitHub offre un sistema di "Issue" e "Milestone" per pianificare e coordinare le attività da svolgere. Tipicamente sarà compito del *Responsabile di Progetto* predisporre le issue relative a una particolare milestone in modo che il resto del team possa avere più autonomia, potendo autoassegnarsi issue ancora aperte. Idealmente le issue saranno collegate con una o più pull request.

Discord

Principale strumento di **comunicazione interna sincrona** e **asincrona**. Vengono utilizzati 3 categorie di canali:

- **Canali Risorse**: condivisione di risorse, creazione di sondaggi per effettuare decisioni problematiche, integrazione con strumenti esterni per permettere notifiche (e.g. GitHub);
- **Canali Testuali**: comunicazioni testuali sincrone e asincrone tra i membri del gruppo;
- **Canali Vocali**: comunicazioni vocali tra i membri del gruppo, con possibilità di condividere lo schermo se necessario.

Ciascuna categoria può contenere un numero variabile di canali, a seconda delle necessità del periodo.

WhatsApp

Principale strumento di **comunicazione interna testuale asincrona**. Viene utilizzato in due modalità:

- **Gruppo**: chat condivisa utilizzata, con parsimonia, per comunicazioni rivolte a tutti i membri;
- **Individuale**: ogni membro del gruppo può essere contattato singolarmente.

Google Calendar

Calendario condiviso del gruppo utilizzato per comunicare e ricordare:

- **Meeting Interni**: di cui saranno specificati:
  - Orario di inizio;
  - Moderatore;
  - Scriba (redazione verbale);
  - Argomenti: specifiche idee da trattare durante una riunione del gruppo.
- **Meeting Esterni**: con proponente o committente;
- **Scadenze Interne**;
- **Scadenze Esterne**;
- Qualsiasi altra attività o evento che può essere collocato in un tempo specifico.

Mantenere il calendario aggiornato è compito del *Responsabile di Progetto*.

Google Drive

Strumento utilizzato come:

- **Directory condivisa** dai membri del gruppo per documenti temporanei o non ufficiali;
- Accesso alla **suite Google**: Docs, Sheets, Slides.

Zoom

Strumento di videochiamata utilizzato principalmente per la comunicazione esterna con committente e proponente.

Google Mail

Utilizzo dell'indirizzo e-mail condiviso `merlunipd@gmail.com` per le **comunicazioni esterne** come gruppo con il proponente e il committente.

4.3 Miglioramento

Durante lo svolgimento dei documenti e del prodotto software cercheremo un miglioramento continuo del nostro lavoro, con l'obiettivo di non ripere errori già fatti, fornendo soluzioni. Per svolgere al meglio questo processo possiamo usufruire del *Piano di Qualifica V1.0.0*, in particolare della sezione *Valutazioni per il miglioramento*, dove è possibile trovare i problemi principali riscontrati dal gruppo, con opportune descrizioni e soluzioni. La valutazione problemi si divide nelle seguenti categorie:

- Organizzazione;
- Ruoli;
- Strumenti di lavoro.

4.4 Formazione

Per garantire un andamento organizzato, simultaneo e alla pari, senza lasciare indietro nessuno, e che quindi favorisca un miglior lavoro asincrono, ogni componente del gruppo, in caso di lacune, dovrà studiarsi in autonomia gli strumenti e le tecnologie utilizzate per documentare e sviluppare il progetto, oppure condividere eventuali conoscenze con gli altri membri per velocizzare questo processo di formazione.
Di seguito sono riportati gli strumenti e le tecnologie utilizzate, con i principali riferimenti usati dal gruppo:

- LaTeX:
  https://www.overleaf.com/learn;
- Git:
  https://docs.github.com/en/get-started/using-git/about-git;
- GitHub:
  https://docs.github.com;
- GitHub Flow:
  https://docs.github.com/en/get-started/quickstart/github-flow.