Indice

1 Progettazione

Dopo aver terminato la fase di Analisi si passerà a quella di Progettazione che vede come protagonisti i progettisti, durante la quale questi devono trovare una soluzione soddisfacente al problema, definire un'architettura logica e i vari diagrammi che la rappresentano. La progettazione permette di:

- Ottimizzare l'uso delle risorse
- Garantire la qualità del prodotto sviluppato
- Suddividere il problema principale in tanti sotto problemi di complessità minore

1.1 Architettura logica

Bisogna definire un'architettura logica del prodotto che dovrà:

- Soddisfare i requisiti definiti nel documento di Analisi dei Requisiti v.1.0.0;
- Essere sicura in caso di malfunzionamenti o intrusioni
- Essere modulare e formato da componenti riutilizzabili;
- Essere modulare e formato da componenti riutilizzabili;
- Essere affidabile;
- Essere comprensibile per future manutenzioni

1.2 Diagrammi UML

Con l'obiettivo di rendere chiare le soluzioni progettuali utilizzate, è necessario l'utilizzo di diagrammi UML. Quest'ultimi devono essere realizzati utilizzando lo standard 2.0.

È richiesta la realizzazione di:

- Diagrammi delle attività: descrivono un processo o un algoritmo;
- Diagrammi dei package;
- Diagrammi di sequenza: rappresentano una sequenza di processi o funzioni;
- Diagrammi di classi: rappresentano le classi utilizzate e le loro relazioni;

In caso vengano utilizzati dei Design Pattern sarà necessario accompagnarli con una descrizione ed un diagramma UML.

1.3 Design Pattern

I progettisti devono utilizzare il design pattern che ritengono più adatto al contesto per rendere l'applicazione più sicura ed efficiente possibile.

Ogni utilizzo di design pattern deve essere brevemente descritto ed accompagnato da un diagramma UML che ne esemplifica il funzionamento.

2 Codifica

In questa sezione vengono descritte le norme che i programmatori devono seguire con l'obiettivo di scrivere codice leggibile, affidabile e mantenibile.

Questa sezione verrà aggiornata durante la fase di progettazione

3 Processi di supporto

3.1 Documentazione

3.1.1 Scopo

Lo scopo di questo processo è quello di redigere e mantenere la documentazione durante tutto il ciclo di vita del software I documenti formali che verranno presentati saranno suddivisi in due categorie:

- **Interni**
  - **Norme di progetto**: lo scopo del documento è di stabilire convenzioni e norme che il team SWEight dovrà seguire durante tutto il progetto

  Studio di fattibilità: il documento presenterà il processo che ha portato il team alla scelta del capitolato, riportando le motivazioni di preferenza o rifiuto di ogni capitolato.
- **Esterni**
  - **Piano di progetto**: documento per l'analisi e la pianificazione della gestione delle risorse di tempo e umane;
  - **Piano di qualifica**: documento che descrive standard e obiettivi che il gruppo deve raggiungere per garantire la qualità di processo e prodotto;
  - **Glossario**: documento che raccoglie tutte le definizioni dei termini poco frequenti utilizzati in tutti i documenti del progetto.
  - **Analisi dei requisiti**: documento che contiene l'analisi dei casi d'uso del prodotto e i diagrammi di interazioni previste con l'utente

3.1.2 Template

Per semplificare e standardizzare la stesura dei documenti previsti è stato creato un template LATEX contenente tutte le impostazioni per la struttura grafica e lo stile di formattazione. Ogni membro del team deve obbligatoriamente utilizzare questo template.

3.1.3 Versioni

Per ogni documento deve essere specificata obbligatoriamente la versione nella seguente forma:
Dove X, Y, e Z sono interi non negativi, e NON DEVONO contenere zeri iniziali. X è la versione major, Y è la versione minor, e Z è la versione patch. Ogni elemento DEVE incrementare come numero a sé. Per esempio: 1.9.0 -> 1.10.0 -> 1.11.0. . . .

- La versione Major zero (0.y.z) è per lo sviluppo iniziale
- X: la versione Major (X.y.z | X > 0) identifica la versione di rilascio. Deve essere incrementata se è stata introdotta qualsiasi modifica non retrocompatibile. Le versioni Patch e Minor devono essere reimpostate a 0 quando la versione Major è incrementata.
- Y: la versione Minor (x.Y.z | x > 0) deve essere incrementata se è stata introdotta una nuova funzionalità. La versione Patch deve essere reimpostata a 0 quando la versione Minor è incrementata.
- Z: la versione Patch (x.y.Z | x > 0) deve essere incrementata solo se sono state introdotte correzioni retrocompatibili di bug. Una correzione di un bug è definita come una modifica interna che corregge un comportamento errato

Una volta che un pacchetto versionato è stato rilasciato, i contenuti di quella versione non devono essere modificati. Qualsiasi modifica deve essere rilasciata come una nuova versione.

3.1.4 Struttura

Ogni documento deve essere realizzato a partire dal template descritto precedentemente, la cui struttura non deve essere modificata, ad eccezione dei verbali e della lettera di presentazione, dovrà essere composta da:

- **Frontespizio**: questa sezione si trova nella prima pagina di ogni documento e deve contener
  * **Logo**:logo del gruppo
  * **Titolo**:titolo del documento
  * **Nome del gruppo**;
  * **Nome del progetto**;
  * **E-mail**:indirizzo di posta elettronica in comune per le comunicazini interne ed esterne;
  * **Versione**: versione del documento;
  * **Owner**:responsabile del documento;
  * **Redattori**: redattori del documento;
  * **Verificatori**: verificatori del documento;
  * **Uso**:nterno o esterno;
  * **Distribuzione**:destinatari del documento;
  * **Descrizione**: breve descrizione del documento;
- **Registro delle modifiche**: questo deve essere riportato nella seconda pagina sotto forma di tabella, contenente: versione del documento, data si salvataggio, descrizione della nuova versione, nominativo e ruolo;
- **Indice**: contiene il nome di tutti i capitoli, sezioni e sottosezioni seguiti dal numero della pagina iniziale. Nel caso in cui il documento contenga immagini e/o tabelle devono essere presenti anche i relativi indici
- **Indice delle immagin**;
- **Indice delle tabelle**;
- **Introduzione**contiene lo scopo del documento, una breve descrizione ed i vari riferimenti normativi e informativi utili al lettore;
- **Contenuto del documento**.

3.1.5 Formattazione delle pagine

Ogni pagina, escluso il frontespizio ed indice, devono contenere:

- **Intestazione**:
  * Logo del gruppo (a sinistra);
  * Numero del capitolo corrente seguito dal nome (a destra);
- **Piè di pagina**:
  * Nome e versione del documento (sinistra)
  *

  Numero pagina, in formato “N di M” dove N è la pagina corrente ed M è il numero di pagine totali.

3.1.6 Norme Tipografiche

- **Maiuscolo**: l'uso del maiuscolo è obbligatorio nei seguenti casi:
  * All'inizio di ogni elemento di un elenco puntato;
  * All'inizio di ogni elemento di un elenco puntato;
  * Nella scrittura degli acronimi;
  * Dopo il punto, punto di domanda, punto esclamativo;
  * Per i ruoli di progetto, i nomi dei documenti, le fasi di progetto, revisioni di progetto oltre a dove previsto dalla lingua italiana.
  * Per i ruoli di progetto, i nomi dei documenti, le fasi di progetto, revisioni di progetto oltre a dove previsto dalla lingua italiana.
- **Grassetto**: il grassetto può essere utilizzato solo per evidenziale l'oggetto trattato di un elenco puntato e per le parole chiave;
- **Collegamenti**: tutti i collegamenti devono essere di colore rosso;
- **Riferimenti a sezioni**: i riferimenti interni al documento devono riportare il numero della sezione, preceduto dal simbolo di paragrafo (esempio §2.1.3)
- **Corsivo**:il corsivo deve essere utilizzato nei seguenti casi:
  * **Ruoli**: in ogni nome di ruolo;
  * **Documenti**: in ogni nome di documento;
  * **Citazion**: in ogni citazione;
- **Codice**: i frammenti di codice devono essere racchiusi in un riquadro di colore nero.
- **Glossario**: per contrassegnare che una parola che è presente nel glossario è necessario aggiungere una G maiuscola al pedice della parola (ad esempio: termine$_{G}$
- **Elenchi puntati**:
  * Ogni elemento dell'elenco deve terminare con il punto e virgola, ad eccezione dell'ultimo elemento che deve terminare con il punto;
  * Gli elementi di primo livello devono avere come stile un pallino pieno nero, quelli di secondo livello un pallino nero vuoto, ad eccezione degli elenchi numerati che devono essere usati solo se è necessario descrivere una sequenza
  * Gli elementi di livello successivo al secondo devono alternare lo stile del primo e del secondo livello;
- **Citazioni**: ogni citazione deve essere accompagnata dal riferimento bibliografico.

3.1.7 Formati

Date: ogni data deve seguire lo standard internazionale per date e orari ISO 8601:2004:
dove:

- AAAA: rappresenta il formato dell'anno scritto con quattro cifre;
- MM: rappresenta il formato del mese scritto con due cifre;
- GG: rappresenta il formato del giorno scritto con due cifre.

3.1.8 Tabelle

Tutte le tabelle devono avere un indice univoco che la identifichi all'interno del documento ed una breve descrizione posizionata sotto di essa. Le intestazioni delle tabelle devono essere in grassetto.

3.1.9 Immagini

Tutte le immagini inserite nei vari documenti devono avere un eguale e sufficiente margine orizzontale e verticale in modo da non ridurre la leggibilità del testo. Ogni immagine deve anch'essa avere un indice univoco che la contraddistingua nell'intero documento.

3.1.10 Diagrammi UML

Il software che deve essere utilizzato per la creazione dei diagrammi UML è Astah versione 8.0.0, gli analisti devono utilizzare la documentazione del software per il suo studio e utilizzo. Ogni diagramma di caso d'uso è identificato in alto a sinistra dal suo codice identificativo e dal nome.

3.1.11 Stesura dei documenti

L'intera documentazione deve essere prodotta utilizzando il linguaggio di markup LATEX, scelto dal team per i seguenti motivi:

- Gestisce elegantemente e con facilità gli indici, i pedici, i riferimenti ed il glossario;
- È un linguaggio che supporta il Versionamento;
- Permette la stesura del documento suddividendolo in tanti sotto progetti che rappresentano le varie sezioni, aiutandoti a gestire meglio il documento;

Per la stesura del documento è consigliato l'utilizzo di Textmaker, un editor che implementa anche un ottimo correttore ortografico e si sillabazione.

3.1.12 Controllo ortografico

Al termine della stesura dei documenti, prima di alla fase di verifica del documento, l'owner deve effettuare un test sulla piattaforma alla pagina: http://www.corrige.it, quest'ultima restituirà un'eventuale lista di errori ortografici, errori di sintassi e l'indice Gulpeas

3.2 Ciclo di vita

Per gestire il ciclo di vita dei documenti, deve essere utilizzato il servizio di issues offerto da GitHub.

1. Per iniziare la stesura di un documento l'owner deve assegnarsi una issues (To do);
2. Quando inizia la stesura del documento (In progress)
   (a) Sposta la issue nella sezione "in progress";
   (b) Crea un branch per lavorare sulla issue.
3. Finita l'attività di scrittura crea un pull request con (Needs review)
   (a) Sé stesso come assigneer;
   (b) I verificatori come reviewers;
   (c) I campi project e milestone vuoti;

4. I verificatori controllano il documento e possono:
   (a) Accettarlo (Reviewer approved);
   (b) Non accettarlo (In progress);
5. Se non accettato, l'owner deve apportare le modifiche descritte dal verificatore
6. Se accettato, sarà compito del responsabile di progetto di effettuare il merge e chiudere la issues (Done)

4 Verifica

4.1 Scopo

La verifica dei processi, documenti e prodotti è un'attività da eseguire continuamente durante lo sviluppo del progetto. Di conseguenza, servono modalità operative chiare e dettagliate per i Verificatori, in modo da uniformare le attività di verifica svolte ed ottenere il miglior risultato possibile.

La corretta implementazione del processo deve:

- Fornire le procedure di verifica necessarie;
- Individuare i criteri per la verifica;
- Individuare eventuali difetti perché possano essere corretti.

4.1.1 Analisi

4.1.2 Analisi statica

- **Walkthrough**: lettura completa del codice sorgente da analizzare. Va utilizzata unicamente durante le prime fasi del progetto in quanto risulta onerosa e non efficiente, questa tecnica di analisi prevede una lettura critica del codice o del documento prodotto. Gli Analisti che la utilizzano devono stilare una lista di controllo con gli errori rilevati più frequentemente;
- **Inspection**: lettura mirata del codice sorgente da analizzare. Questa tecnica di analisi presuppone l'esperienza da parte del verificatore nell'individuare gli errori e le anomalie più frequenti in modo tale da creare una lista di controllo per poter localizzare eventuali punti critici in cui cercare errori. Dopo ogni analisi la lista di controllo deve essere incrementata con eventuali nuovi errori rilevati.

4.1.3 Analisi dinamica

Si applica solo alle componenti software e consiste nella verifica e validazione attraverso i test. Per garantire la correttezza è necessario che i test siano ripetibili, dato lo stesso input il test deve produrre sempre lo stesso output.

Solo i test con queste caratteristiche riescono a verificare la correttezza del prodotto

Per ogni test deve quindi essere definito:

- Ambiente: sistema hardware e software in cui viene eseguito il test;
- Stato iniziale: stato iniziale da cui si parte ad eseguire il test;
- Input: input inserito;
- Input: input inserito;

4.1.4 Test

4.1.5 Test di unità

Il test di unità verifica che ogni singola unità funzioni correttamente, le unità sono specificate nella progettazione di dettaglio.

In particolare, si verifica che i requisiti per quella determinata unità siano soddisfatti.

- **Test strutturale (white-box)**: verifica la logica interna del codice dell'unità cercando la massima copertura. Ogni singola prova deve attivare un singolo cammino di esecuzione all'interno dell'unità. L'insieme di dati di ingresso che ottiene quell'effetto costituisce un caso di prova;
- • fa riferimento alla specifica dell'unità utilizza dati di ingresso capaci di provocare l'esito atteso. Ciascun insieme di dati di ingresso che produca un dato comportamento funzionale costituisce un singolo caso di prova.

4.1.6 Test di Integrazione

Si applica alle componenti specificate nella progettazione architetturale, rappresenta l'estensione logica del test di unità. Consiste nella combinazione di due unità già sottoposte a test in un solo componente e nel test dell'interfaccia presente tra le due. Il test di integrazione consente di individuare i problemi che si verificano quando due unità si combinano. Per effettuare tali test si farà uso di classi appositamente create per simulare e verificare l'interazione. Strategie di integrazione:

- **Assemblare parti in modo incrementale**:
  - Si sviluppano e si integrano prima le parti con minore dipendenza funzionale e maggiore utilità;
  - Poi si risale l'albero delle dipendenze;
  - Questa strategia riduce il numero di stub necessari al test ma ritarda la disponibilità di funzionalità di alto livello;
- **Top-down**:
  - Si sviluppano prima le parti più esterne, quelle poste sulle foglie dell'albero delle dipendenze e poi si scende;
  - Questa strategia comporta l'uso di molti stub ma integra a partire dalle funzionalità di più alto livello;
- **Assemblare produttori prima dei consumatori**;
- **Assemblare in modo che ogni passo di integrazione sia reversibile**.

4.1.7 Test di sistema

Il test di sistema consiste nella validazione del sistema ed è un test di tipo funzionale. Viene eseguito quando si ritiene che il prodotto sia giunto ad una versione definitiva, verificando la completa copertura dei requisiti software.

4.1.8 Test di regressione

Il test di regressione va eseguito ogni volta che viene modificata un'implementazione in un programma. È possibile eseguire nuovamente i test esistenti sul codice modificato, integrando solo le parti che abbiano precedentemente superato il test di unità, per stabilire se le modifiche apportate hanno alterato elementi precedentemente funzionanti. Se necessario è anche possibile scrivere nuovi test.
I contenuti del test di regressione vanno decisi nel momento in cui si approvano modifiche al SW

4.1.9 Test di accettazione(collaudo)

Il test di accettazione consiste nel collaudo del sistema eseguito dal committente. Se l'esito è positivo si può procedere al rilascio ufficiale del prodotto.

4.1.10 Verifica dei documenti

Ogni volte che un documento viene modificato, per essere approvato dal Responsabile di progetto, deve essere verificato da un Verificatore.
Per essere certi che il documento sia conforme alle norme di progetto e agli altri documenti presentate, il Verificatore deve:

- Verificare il contenuto: controllare che tutto il contenuto del documento sia inserito nella giusta sezione ed adeguatamente impaginato;
- Verificare il glossario: controllare che tutte le parole da inserire nel glossario siano state inserite e che tutte le parole inserite nel glossario, siano state targate con il pedice "G";
- Verifica tipografica: controllare con il controllo ortografico dell'editor utilizzato eventuali errori e correggerli;
- Segnalare gli errori: una volta completata la verifica, deve segnalare tutti gli errori trovati e comunicarli al redattore.

4.1.11 Verifica dei diagrammi

I diagrammi devono essere verificati manualmente dal Verificatore che deve controllare che aderiscano correttamente allo standard 2.0. In particolare deve controllare che i diagrammi di flusso siano rappresentati in maniera corretta e che i diagrammi utilizzino correttamente le inclusioni e le estensioni.

5 Progressi organizzativi

5.1 Obiettivo

Gli obiettivi del processo di gestione sono:

- Raggiungere l'efficienza e l'efficacia seguendo un approccio sistematico dei processi software al fine di raggiungere gli obiettivi di progetto rispettando le scadenze;
- Introdurre nuovi processi o migliorare quelli già esistenti.

5.1.1 Incontri

5.1.2 Interni

Una riunione dell'intero team è richiesta in caso di necessità del Responsabile di progetto. Nel momento in cui un membro del gruppo abbia la necessità di incontrarsi con il team, deve inviare una richiesta al Responsabile di progetto che la valuta e organizza un eventuale incontro. Questi sono fissati solo per discutere di argomenti attinenti al progetto.

5.1.3 Esterni

Il Responsabile di progetto è l'unico a poter fissare gli incontri con il proponente o i committenti e successivamente informare i componenti del team. Non è necessaria la presenza dell'intero gruppo di lavoro durante gli incontri con gli esterni. Il numero di partecipanti sarà quindi limitato a 2 rappresentanti del team di sviluppo.
Il Responsabile di progetto deve essere sempre presente agli incontri con gli esterni mentre il secondo partecipante del team è scelto in base all'ordine del giorno.
Ogni riunione comprende la stesura di un verbale ufficiale contenente le seguenti informazioni:

- Data e ora
- Luogo
- Partecipanti esterni
- Partecipanti interni
- Ordine del giorno
- Domande e risposte

5.1.4 Comunicazione

5.1.5 Interna

Per le comunicazioni interne al gruppo è stato adottato Slack, un'applicazione di messaggistica multipiattaforma con funzionalità specifiche per gruppi di lavoro.
Su Slack si possono integrare Bot, creare canali tematici per rendere più efficiente lo scambio e il reperimento delle informazioni, fare ricerche di messaggi vecchi e condividere file di grosse dimensioni.
Si è deciso di affiancare discord quando risulti necessaria una comunicazione immediata ma sia impossibile riunirsi di persona (es. vacanze di natale).
In Slack, sono stati creati dei canali per dividere le comunicazioni con temi comuni:

- Analisti;
- Colletta;
- Norme-di-progetto;
- Random;
- Verifica;
- Link.

5.1.6 Esterna

Per le comunicazioni esterne è stata creata una casella di posta elettronica .
Tale indirizzo deve essere l'unico canale di comunicazione esistente tra il gruppo di lavoro e l'esterno.
Come descritto nel capitolato d' appalto:

- La proponente può essere contattata in qualsiasi momento, solo dal responsabile di progetto, attraverso l'indirizzo e-mail , al quale risponde il reparto tecnologico dell'azienda;
- Le e-mail devono contenere in oggetto la sigla "UNIPD-SWE" e dovranno essere indirizzate all'attenzione di Giulio Paci, che sarà il referente principale;
- Ogni e-mail ricevuta su questo indirizzo viene inoltrata automaticamente alla casella personale di ciascun membro, tramite l'utilizzo di filtri di Gmail;
- In alternativa per una comunicazione ritenute indifferibili è possibile telefonare al numero 0490998335.

5.2 Gestione delle responsabilità e ruoli

5.2.1 Introduzione

I ruoli di progetto rappresentano le figure professionali che lavorano al progetto. Ogni membro del gruppo, in un dato momento, dovrà ricoprire un determinato ruolo.
La rotazione è opzionale e si terrà conto delle preferenze personali di ognuno. Ogni membro deve rispettare il proprio ruolo e svolgere le attività assegnategli, secondo quanto stabilito nel Piano di Progetto v 1.0.0. Si avrà l'accortezza di evitare situazioni di conflitto di interesse come, ad esempio, essere verificatori di documenti prodotti da sè stessi, compromettendone cosi la qualità.

5.2.2 Responsabile di progetto

Il Responsabile di Progetto, o Project Manager, è una figura necessaria alla gestione dell'intero progetto. Egli raccoglie su di sé le responsabilità decisionali di scelta e approvazione e costituisce il centro di coordinamento per l'intero progetto; in particolare, rappresenta il gruppo di lavoro nei confronti dei Committenti e della Proponente.
In particolare, ha l'incarico di:

- Organizzare incontri interni ed esterni;
- Pianificare le attività svolte dal gruppo;
- Individuare per ciascun compito un membro del gruppo per svolgerlo;
- Analizzare, monitorare e gestire i rischi.

5.2.3 Amministratore di progetto

L'Amministratore di Progetto è la figura professionale che si deve gestire l'ambiente di lavoro, al fine di aumentare l'efficienza e portare qualità.
Ha l'incarico di:

- Ricercare nuovi strumenti che migliorino l'efficienza;
- Gestire la documentazione di progetto;
- Occuparsi del controllo di versione del prodotto;
- Occuparsi della configurazione del prodotto.

5.2.4 Analista

L'Analista è la figura che svolge le attività di analisi al fine di comprendere appieno il dominio del problema.
Ha l'incarico di:

- Analizzare i requisiti del prodotto;
- Analizzare i requisiti di dominio;
- Redigere il documento Studio di Fattibilità;
- Redigere il documento Analisi dei Requisiti.

5.2.5 Progettista

Il Progettista è il responsabile delle scelte architetturali del progetto e ne influenza gli aspetti tecnici e tecnologici.
Partendo dalle attività dell'Analista, il Progettista ha il compito di trovare una possibile soluzione per i problemi e i requisiti precedentemente individuati.
Ha l'incarico di:

- Comprendere a fondo i requisiti nel documento Analisi dei Requisiti;
- Comprendere a fondo i requisiti nel documento Analisi dei Requisiti;
- Redigere la documentazione tecnica per il prodotto software;
- Redigere il documento Manuale Sviluppatore.

5.2.6 Programmatore

Il Programmatore è la figura che provvederà alla codifica della soluzione, studiata e spiegata dal Progettista.
Ha l'incarico di:

- Scrivere il codice del prodotto software che rispetti le decisioni del Progettista;
- Redigere il documento Manuale Utente.

5.2.7 Verificatore

Il Verificatore è una figura presente per l'intero ciclo di vita del software e controlla che le attività svolte siano conformi alle attese e alle norme prestabilite.
Ha l'incarico di:

- Verificare che ciascuna attività svolta sia conforme alle norme stabilite nel progetto;
- Controllare che, per ogni stadio del ciclo di vita del prodotto, questo sia conforme al Piano di Qualifica.

5.3 Pianificazione

Il Responsabile decide le scadenze tenendo conto degli impegni lavorativi e scolastici di ogni membro. Stima i costi e le risorse necessarie, pianifica le attività e le assegna alle persone.
Il responsabile di progetto deve definire il piano di progetto, in cui descrive attività e compiti utili/necessari all' esecuzione di processo.
Il piano deve:

- Fissare un tempo di completamento dei compiti;
- Dare una stima del tempo richiesto dei compiti;
- Adeguare le risorse necessarie per eseguire i compiti;
- Allocare i compiti;
- Assegnare le responsabilità;
- Analizzare i rischi;
- Definire delle metriche per il controllo della qualità;
- Associare dei costi al processo di esecuzione;
- Fornire un'infrastruttura;

5.4 Monitoraggio del piano

Il Responsabile di Progetto supervisiona l'esecuzione del progetto fornendo report interni sulla progressione del processo.
Egli deve investigare, analizzare e risolvere i problemi scoperti durante l'esecuzione ed eventualmente può rivedere la pianificazione temporale per far fronte a tali problemi e a cambi di strategia.

- I report redatti, in conclusione di ogni periodo, confluiranno nel Piano di Qualifica.
- Al termine del periodo di riferimento, il Responsabile di Progetto deve confrontare i risultati ottenuti con gli obiettivi prefissati e valutare strumenti, attività e processi impiegati per il completamento dei processi.
- La valutazione finale e il tracciamento di strumenti e tecnologie dovranno confluire nel Piano di Qualifica, eventuali rischi e piani di contingenza utilizzati dovranno essere riportati nel Piano di Progetto.

5.5 Gestione dell'infrastruttura

5.5.1 Introduzione

Inizializzare, implementare e gestire processi software, richiede l'istanziazione di un'infrastruttura ad hoc per l'intero progetto. In questa sezione verranno verrà normato e descritto l'ambiente di lavoro e gli strumenti o software utilizzati.

5.5.2 Versionamento

5.5.3 Introduzione

Git, il più usato e conosciuto software di controllo di versione open source, è stato scelto per il controllo di versione, GitHub invece è la piattaforma di web hosting.
Il repository dedicato ai documenti si trova all'indirizzo: `https://github.com/SWEightgroup/Colletta` ed è formata dalle seguenti subdirectory:

- RR contiene tutti i documenti da consegnare per la Revisione dei Requisiti;
- RP contiene tutti i documenti da consegnare per la Revisione di Progettazione;
- RQ contiene tutti i documenti da consegnare per la Revisione di Qualifica;
- RA contiene tutti i documenti da consegnare per la Revisione di Accettazione.

5.5.4 Strumenti di condivisione

Un altro servizio utilizzato dal team è Google Drive, servizio web in ambiente cloud, di memorizzazione e sincronizzazione online. Questo servizio ha anche un tool aggiuntivo per l'integrazione Slack, che permette al gruppo un rapido scambio di documenti.

5.6 Miglioramento continuo dei processi

Processi, attività o compiti istanziati possono non essere definitivi. Se al termine di una delle fasi di progetto, un membro del gruppo dovesse trovare difficoltà, oppure se trovasse procedure più efficienti ed efficaci di quelle in uso, può decidere di proporre al responsabile di progetto di istanziarne di nuovi o modificare quelli già esistenti.
Il responsabile di progetto organizzerà un incontro con tutti i membri di SWEight e procederà ad esporre i nuovi spunti suggeriti, ascoltando i pareri di tutti.

La decisione finale spetta comunque al responsabile, che dopo aver valutato l'impatto di qualsiasi modifica al piano ed aver controllato che ciò non comporti l'impossibilità nel raggiungere gli obiettivi prefissati nei tempi stabiliti, potrà decidere di attuarla.

5.7 Training process

È il processo che fornisce e mantiene la formazione del personale. Acquisizione, supporto, sviluppo esecuzione, mantenimento del prodotto software sono largamente dipendenti dalle conoscenze e dalle capacità dei membri del gruppo.
Per questo ognuno deve procedere in modo autonomo con lo studio individuale delle tecnologie che verranno utilizzate nel corso del progetto, prendendo come riferimento, oltre al materiale indicato nella sottosezione Riferimenti normativi, anche la seguente documentazione:

- Latex: `https://www.latex-project.org`;
- GitHub: `https://guides.github.com`;
- Slack: `https://get.slack.help/hc/en-us`;
- Astah: `http://astah.net/manual`.