



Norme di Progetto

Informazioni sul documento

| | |
|----------------------------|--|
| Nome documento | Norme di Progetto |
| Versione | v2.0.0 |
| Data redazione | 2013-12-02 |
| Redattori | <ul style="list-style-type: none">• Adami Alberto• Bissacco Nicolò |
| Verificatori | <ul style="list-style-type: none">• Martignago Jimmy |
| Approvazione | <ul style="list-style-type: none">• Feltre Beatrice• <i>Seven Monkeys</i> |
| Lista distribuzione | <ul style="list-style-type: none">• <i>Prof. Tullio Vardanega</i>• <i>Prof. Riccardo Cardin</i> |
| Uso | Interno |

Sommario

Documento contenente l'insieme di norme stabilite dal gruppo *Seven Monkeys* per la realizzazione di Romeo.

Diario delle Modifiche

| Modifica | Autore & Ruolo | Data | Versione |
|---|--|------------|----------|
| <i>Approvazione del documento</i> | Adami Alberto <i>Responsabile di Progetto</i> | 2014-02-03 | v2.0.0 |
| <i>Verifica del documento</i> | Bissacco Nicolò <i>Verificatore</i> | 2014-01-31 | v1.3.0 |
| <i>Apportate modifiche a seguito della verifica</i> | Luisetto Luca <i>Amministratore di Progetto</i> | 2014-01-26 | v1.2.1 |
| <i>Verifica del documento</i> | Bissacco Nicolò <i>Verificatore</i> | 2014-01-24 | v1.2.0 |
| <i>Incremento sezione 7.4.4</i> | Feltre Beatrice <i>Amministratore di Progetto</i> | 2014-01-20 | v1.1.4 |
| <i>Incremento sezione Ambiente di lavoro</i> | Feltre Beatrice <i>Amministratore di Progetto</i> | 2014-01-14 | v1.1.3 |
| <i>Correzione degli errori rilevati nella Revisione dei Requisiti</i> | Luisetto Luca <i>Amministratore di Progetto</i> | 2014-01-13 | v1.1.2 |
| <i>Cambiamenti alla struttura del documento in base ai suggerimenti del Committente</i> | Luisetto Luca <i>Amministratore di Progetto</i> | 2014-01-10 | v1.1.1 |
| <i>Eseguita verifica del documento</i> | Luisetto Luca <i>Verificatore</i> | 2014-01-07 | v1.1.0 |
| <i>Incremento sezione Ambiente di lavoro</i> | Feltre Beatrice <i>Amministratore di Progetto</i> | 2014-12-27 | v1.0.1 |
| <i>Approvazione del documento</i> | Feltre Beatrice <i>Responsabile di Progetto</i> | 2013-12-17 | v1.0.0 |
| <i>Eseguita verifica del documento</i> | Martignago Jimmy <i>Verificatore</i> | 2013-12-16 | v0.4.0 |
| <i>Aggiunte modifiche a seguito verifica</i> | Adami Alberto <i>Amministratore di Progetto</i> | 2013-12-15 | v0.3.1 |
| <i>Eseguita verifica del documento</i> | Martignago Jimmy <i>Verificatore</i> | 2013-12-14 | v0.3.0 |
| <i>Stesura sezioni progettazione e verifica</i> | Adami Alberto <i>Amministratore di Progetto</i> | 2013-12-13 | v0.2.3 |
| <i>Stesura sezione analisi dei requisiti</i> | Bissacco Nicolò <i>Amministratore di Progetto</i> | 2013-12-12 | v0.2.2 |
| <i>Stesura sezione ambiente di lavoro</i> | Adami Alberto <i>Amministratore di Progetto</i> | 2013-12-11 | v0.2.1 |
| <i>Eseguita verifica del documento</i> | Martignago Jimmy <i>Verificatore</i> | 2013-12-07 | v0.2.0 |
| <i>Aggiunte modifiche a seguito verifica</i> | Bissacco Nicolò <i>Amministratore di Progetto</i> | 2013-12-07 | v0.1.1 |
| <i>Eseguita verifica del documento</i> | Martignago Jimmy <i>Verificatore</i> | 2013-12-06 | v0.1.0 |
| <i>Aggiunta sezione glossario</i> | Adami Alberto <i>Amministratore di Progetto</i> | 2013-12-06 | v0.0.5 |
| <i>Stesura sezione protocollo</i> | Bissacco Nicolò <i>Amministratore di Progetto</i> | 2013-12-05 | v0.0.4 |
| <i>Aggiunta delle sezioni documenti e codifica e convenzioni</i> | Adami Alberto <i>Amministratore di Progetto</i> | 2013-12-04 | v0.0.3 |

| | | | |
|--|--|------------|--------|
| <i>Stesura delle sezioni comunicazioni e repository</i> | Bissacco Nicolò <i>Amministratore di Progetto</i> | 2013-12-03 | v0.0.2 |
| <i>Inizio stesura del documento e stesura delle sezioni introduzione e incontri.</i> | Adami Alberto <i>Amministratore di Progetto</i> | 2013-12-02 | v0.0.1 |

Indice

| | | |
|----------|---------------------------------|-----------|
| 1 | Introduzione | 1 |
| 1.1 | Scopo del documento | 1 |
| 1.2 | Glossario | 1 |
| 1.3 | Riferimenti | 1 |
| 1.3.1 | Normativi | 1 |
| 1.3.2 | Informativi | 1 |
| 2 | Comunicazioni | 2 |
| 2.1 | Comunicazioni esterne | 2 |
| 2.2 | Comunicazioni interne | 2 |
| 2.3 | Composizione email | 2 |
| 2.3.1 | Mittente | 2 |
| 2.3.2 | Destinatario | 2 |
| 2.3.3 | Oggetto | 3 |
| 2.3.4 | Corpo | 3 |
| 2.3.5 | Allegati | 3 |
| 3 | Incontri | 4 |
| 3.1 | Incontri esterni | 4 |
| 3.2 | Incontri interni | 4 |
| 3.3 | Richieste di incontri | 4 |
| 4 | Repository | 5 |
| 4.1 | Repository della documentazione | 5 |
| 4.2 | Repository del codice | 6 |
| 5 | Documenti | 7 |
| 5.1 | Versionamento | 7 |
| 5.2 | Template | 7 |
| 5.3 | Struttura dei documenti | 7 |
| 5.3.1 | Prima pagina | 7 |
| 5.3.2 | Diario delle modifiche | 8 |
| 5.3.3 | Indici | 8 |
| 5.3.4 | Formattazione di una pagina | 8 |
| 5.4 | Classificazione dei documenti | 8 |
| 5.4.1 | Documenti informali | 8 |
| 5.4.2 | Documenti formali | 9 |
| 5.4.3 | Verbali | 9 |
| 5.5 | Norme tipografiche | 9 |
| 5.5.1 | Stile di testo | 10 |
| 5.5.2 | Punteggiatura | 10 |
| 5.5.3 | Composizione del testo | 10 |
| 5.5.4 | Formati ricorrenti | 11 |
| 5.6 | Componenti grafiche | 11 |
| 5.6.1 | Immagini | 11 |
| 5.6.2 | Tabelle | 12 |
| 5.7 | Glossario | 12 |
| 6 | Procedure e regole | 13 |

| | | |
|----------|--|-----------|
| 6.1 | Creazione di un documento | 13 |
| 6.2 | Avanzamento di un documento | 14 |
| 6.3 | Interazione con il repository | 14 |
| 6.4 | Gestione del Glossario | 15 |
| 6.4.1 | Inserimento di un termine | 15 |
| 6.4.2 | Eliminazione di un termine | 16 |
| 6.5 | Rotazione dei ruoli | 17 |
| 7 | Processi | 18 |
| 7.1 | Gestione di progetto | 18 |
| 7.1.1 | Pianificare attività | 18 |
| 7.1.2 | Gestione delle risorse | 18 |
| 7.1.3 | Analisi e prevenzione dei rischi | 18 |
| 7.2 | Analisi dei Requisiti | 18 |
| 7.2.1 | Studio di Fattibilità | 18 |
| 7.2.2 | Analisi dei Requisiti | 19 |
| 7.3 | Progettazione | 20 |
| 7.3.1 | Specifica Tecnica | 20 |
| 7.3.2 | Definizione di Prodotto | 20 |
| 7.4 | Verifica | 21 |
| 7.4.1 | Tecniche di analisi | 21 |
| 7.4.2 | Verifica dei documenti | 22 |
| 7.4.3 | Verifica dei diagrammi | 22 |
| 7.4.4 | Verifica del codice | 22 |
| 7.5 | Codifica | 23 |
| 7.5.1 | Linguaggio di codifica | 24 |
| 7.5.2 | Convenzioni di codifica | 24 |
| 8 | Protocollo per la gestione del progetto | 26 |
| 8.1 | Creazione milestone | 26 |
| 8.2 | Creazione ticket | 27 |
| 8.3 | Esecuzione dei compiti | 29 |
| 8.4 | Chiusura della milestone | 29 |
| 9 | Ambiente di Lavoro | 30 |
| 9.1 | Sistema operativo | 30 |
| 9.2 | Coordinamento | 30 |
| 9.2.1 | Repository Git | 30 |
| 9.2.2 | Dropbox | 30 |
| 9.2.3 | Google Drive | 31 |
| 9.2.4 | Google Calendar | 31 |
| 9.2.5 | Jenkins | 31 |
| 9.3 | Pianificazione delle attività | 31 |
| 9.4 | Tracciamento bug | 31 |
| 9.5 | ReqMonkeys | 32 |
| 9.5.1 | Analisi dei Requisiti | 32 |
| 9.6 | Strumenti per i documenti | 32 |
| 9.6.1 | L ^A T _E X | 32 |
| 9.6.2 | Script | 32 |
| 9.6.3 | Controllo ortografico | 33 |
| 9.6.4 | Diagrammi UML | 33 |

| | | |
|----------|-------------------------------------|-----------|
| 9.7 | Strumenti per lo sviluppo | 33 |
| 9.7.1 | Framework | 33 |
| 9.7.2 | Librerie | 33 |
| 9.7.3 | Ambiente di codifica | 34 |
| 9.7.4 | Documentazione | 34 |
| A | Lista di controllo | 35 |

Elenco delle figure

| | | |
|----|---|----|
| 1 | Organizzazione file system del repository dei documenti. | 5 |
| 2 | Procedura di creazione di un documento | 13 |
| 3 | Procedura di avanzamento di un documento | 14 |
| 4 | Procedura per l'utilizzo del Repository | 15 |
| 5 | Procedimento per l'inserimento di un termine. | 16 |
| 6 | Procedimento per l'eliminazione di un termine | 17 |
| 7 | Interfaccia di GitHub per la creazione di una milestone | 26 |
| 8 | Modello per la gestione di una milestone | 27 |
| 9 | Modello di ticket per la creazione di un compito generico | 28 |
| 10 | Modello di ticket per la creazione di un ticket di verifica | 28 |
| 11 | Interfaccia di GitHub per la creazione di un ticket. | 29 |
| 12 | Schermata di login di mantis | 32 |

1 Introduzione

1.1 Scopo del documento

Questo documento ha lo scopo di fissare le norme che tutti i membri del gruppo *Seven Monkeys* dovranno rispettare durante lo svolgimento del progetto Romeo.

Tutti i membri sono tenuti a leggere il documento e a seguire *rigorosamente* le norme ivi descritte, al fine di garantire un lavoro efficiente, efficace e per avere un'uniformità nei documenti prodotti.

Qualora ve ne sia la necessità, ogni membro del gruppo potrà contattare l'*Amministratore di Progetto* per suggerire l'aggiunta di nuove norme, oppure cambiamenti alle norme già esistenti.

L'*Amministratore di Progetto*, dopo essersi consultato con gli altri membri del gruppo, avrà la facoltà di accettare o rifiutare i suggerimenti proposti.

Le norme esposte nel documento riguarderanno tutti gli aspetti che sono inerenti allo sviluppo del prodotto software.

Il documento, in particolare, pone l'accento sui seguenti punti:

- Le interazioni tra i vari membri del gruppo e verso l'esterno;
- Le modalità di accesso al repository **G**;
- La stesura dei documenti e le varie convenzioni di scrittura utilizzate;
- Le norme utilizzate nella scrittura del codice;
- La definizione dell'ambiente di lavoro.

1.2 Glossario

Al fine di evitare ogni ambiguità e per permettere al lettore una migliore comprensione dei termini e acronimi utilizzati nei vari documenti formali, essi sono riportati nel *Glossario v2.0.0* che contiene una descrizione approfondita di tali termini e acronimi. Ogni volta che compare un termine presente nel *Glossario*, esso è marcato con una "**G**" in pedice.

1.3 Riferimenti

1.3.1 Normativi

- Qt Coding Conventions: <http://qt-project.org/wiki/Coding-Conventions>;
- Qt Coding Style: http://qt-project.org/wiki/Qt_Coding_Style.

1.3.2 Informativi

- *Piano di Progetto v2.0.0*

2 Comunicazioni

2.1 Comunicazioni esterne

Per le comunicazioni verso l'esterno è stato creato un indirizzo di posta elettronica apposito: 7monkeys.swe@gmail.com.

Tale indirizzo dovrà essere l'unico servizio utilizzabile per le comunicazioni verso l'esterno.

Sarà solo il *Responsabile di Progetto* a utilizzare l'indirizzo di posta per conto del gruppo *Seven Monkeys* intrattenendo le corrispondenze con i proponenti e i commitenti.

Eventualmente il *Responsabile di Progetto* provvederà ad inoltrare le conversazioni a tutti i membri del gruppo tramite la mailing list_G, ma solo se ritiene che sia necessario.

2.2 Comunicazioni interne

Le comunicazioni interne verranno eseguite tramite la mailing list_G: 7monkeys.swe@gmail.com.

Quando un membro del gruppo vuole inviare un'email a tutti i componenti, deve inviare il messaggio dalla sua email personale verso l'indirizzo 7monkeys.swe@gmail.com. Un inoltrato automatico provvederà a trasmettere l'email agli indirizzi personali dei componenti del gruppo presenti nella mailing list_G, tranne che al membro che ha inviato il messaggio. In questo modo tutti i componenti saranno sempre al corrente di tutti gli incontri e impegni del gruppo.

Al fine di facilitare le comunicazioni tra i vari membri del gruppo, viene utilizzato Google Hangout_G come servizio di messaggistica istantanea e per le video chiamate.

È necessario redigere un verbale¹ nel caso in cui siano state prese decisioni o siano emersi dettagli inerenti allo sviluppo del progetto.

2.3 Composizione email

In questo paragrafo è descritta la forma che deve avere una email sia per una comunicazione interna che esterna.

2.3.1 Mittente

Il mittente della email potrà cambiare a seconda del tipo di comunicazione svolta:

- **Esterna:** l'unico indirizzo utilizzabile per comunicare verso l'esterno dovrà essere *necessariamente* l'indirizzo 7monkeys.swe@gmail.com, il quale sarà usato *esclusivamente* dal *Responsabile di Progetto*;
- **Interna:** in questo caso andrà messo l'indirizzo personale di chi scrive.

2.3.2 Destinatario

Il destinatario della e-mail cambierà a seconda che si tratti di una comunicazione interna o esterna:

- **Esterna:** l'indirizzo del destinatario potrà variare a seconda che si voglia comunicare con il Prof. Tullio Vardanega, il Prof. Riccardo Cardin o con i proponenti del progetto;
- **Interna:** l'unico indirizzo utilizzabile è 7monkeys.swe@gmail.com.

Sono ammesse alcune eccezioni:

¹Vedi sezione 5.4.3 per maggiori dettagli.

- **Proposta all'Amministratore di Progetto:** nel caso in cui un membro del gruppo voglia contattare l'Amministratore di Progetto per richiedere cambiamenti alle norme, il membro dovrà contattarlo al suo indirizzo di posta personale²;
- **Proposta al Responsabile di Progetto:** nel caso in cui un membro del gruppo voglia richiedere una riunione, dovrà contattarlo al suo indirizzo di posta personale³;
- **Comunicazione ristretta tra alcuni membri del team:** in alcuni casi i membri del team potrebbero avere la necessità di comunicare tra di loro e utilizzeranno i loro indirizzi personali⁴.

2.3.3 Oggetto

L'oggetto deve essere chiaro, esaustivo e possibilmente univoco, in modo da riconoscerlo da quelli precedenti.

Nel caso si debba inviare un messaggio alla mailing list_G, vi è l'obbligo di aggiungere "**CI:**" all'inizio dell'oggetto.

2.3.4 Corpo

Il corpo di un messaggio dovrà avere tutti gli elementi e le informazioni che permettano a tutti i destinatari di capire correttamente l'argomento trattato.

Se alcune parti del messaggio si riferiscono a particolari membri del gruppo o a certi ruoli di progetto si dovrà usare la seguente sintassi: "**@Cognome Nome**" o "**@Nome Ruolo**" per riferirsi ad essi. Nel caso in cui il corpo del messaggio abbia più di trenta righe, è preferibile scrivere un corpo riassuntivo ed allegare un file **PDF_G** che scenda più nel dettaglio.

Alla fine del corpo il mittente dovrà sempre firmarsi col suo cognome, nome e ruolo.

2.3.5 Allegati

Viene consentito di allegare dei file al messaggio, preferibilmente in formato **PDF_G**, i quali non dovranno superare i 20MB.

Essi potranno essere utilizzati ad esempio per allegare il verbale di un incontro.

²Sarà possibile trovare i vari recapiti e-mail personali nel documento "contatti" condiviso su Google Drive.

³Vedi nota precedente.

⁴Vedi due note precedenti.

3 Incontri

Sarà il *Responsabile di Progetto* a fissare gli incontri, sia interni che esterni.

Per ogni incontro dovrà essere specificata data, ora, luogo, ordine del giorno e motivi della riunione; tali informazioni dovranno essere rese disponibili con almeno quattro giorni di anticipo.

Successivamente alla decisione di una nuova data il *Responsabile di Progetto* provvederà alla creazione di un evento sul calendario condiviso su Google Calendar_G.

3.1 Incontri esterni

Sarà il *Responsabile di Progetto* a fissare gli incontri con i proponenti e/o committenti utilizzando la casella di posta creata appositamente⁵. Il *Responsabile di Progetto* prima di prendere alcun accordo con le parti esterne dovrà contattare i vari componenti del gruppo, per sentire se almeno cinque membri concordano.

In caso positivo il *Responsabile di Progetto* provvederà a contattare i proponenti e/o committenti per fissare la data dell'incontro.

Sarà compito del *Responsabile di Progetto* redigere il verbale dell'incontro avvenuto.

3.2 Incontri interni

Sarà il *Responsabile di Progetto* a fissare gli incontri interni, contattando tutti i membri del gruppo. Gli incontri interni dovranno avere una frequenza almeno quindicinale. Ogni componente del gruppo è tenuto a leggere regolarmente la posta elettronica personale e rispondere ad eventuali richieste di un incontro interno. Nel caso in cui i membri disponibili a partecipare alla riunione in una certa data siano meno di quattro, questa verrà posticipata o anticipata, in modo che siano presenti un numero adeguato di persone. Inoltre, è possibile e auspicabile che siano necessarie riunioni tra specifici membri del gruppo, ad esempio: in fase di analisi può essere utile che solo gli *Analisti* si incontrino tra di loro, senza il resto del gruppo. I restanti componenti del gruppo saranno comunque informati sui contenuti e le decisioni prese tramite invio di una e-mail alla mailing list_G o alla pubblicazione di un verbale sul repository_G nel caso siano state prese decisioni importanti.

3.3 Richieste di incontri

Qualora ve ne sia la necessità, ogni membro del gruppo potrà richiedere un incontro sia interno che esterno, contattando personalmente il *Responsabile di Progetto* ed esponendo i motivi della richiesta.

Il *Responsabile di Progetto* avrà potere di scegliere se accettare o rifiutare la richiesta.

⁵Vedi 2.1 per maggiori dettagli.

4 Repository

In questa sezione sono definiti gli strumenti utilizzati per la condivisione e il versionamento dei vari file.

Per la gestione della documentazione e dei file di codifica sono stati creati due repository_G distinti. I repository_G sono privati e quindi accessibili solo ai membri del gruppo *Seven Monkeys*.

I repository_G sono ospitati dal servizio GitHub_G, il quale utilizza il sistema di versionamento Git_G.

4.1 Repository della documentazione

L'URL del repository_G della documentazione è: <https://github.com/nicolobissacco/7MonkeysDoc>.

All'interno di questo repository_G verranno memorizzati tutti i file necessari alla generazione dei documenti. Di seguito verrà descritta sinteticamente l'organizzazione interna del filesystem.

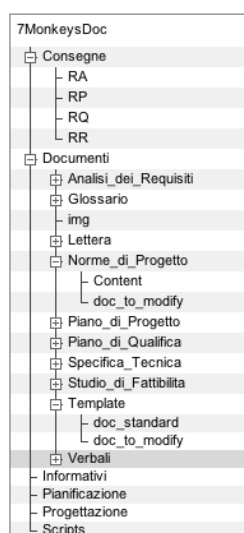


Figura 1: Organizzazione file system del repository dei documenti.

- **Consegne:** contiene tutti i documenti in formato PDF_G consegnati nelle varie revisioni. I file sono raggruppati in sotto cartelle denominate con l'acronimo della revisione a cui fanno riferimento. La struttura delle sottocartelle sarà la seguente:
 - **RR:** Revisione dei Requisiti;
 - **RP:** Revisione di Progettazione;
 - **RQ:** Revisione di Qualifica;
 - **RA:** Revisione di Accettazione.
- **Documenti:** è la cartella contenente i sorgenti di tutti i documenti, sia in stato di redazione, sia completati. Il suo contenuto è organizzato nel seguente modo:
 - **Nome_del_Documento:** ogni documento in fase di redazione dovrà avere la propria cartella, denominata con lo stesso nome del documento (vedi 1). Se il documento necessita di immagini, esse dovranno essere salvate in una sottocartella denominata "Immagini";
 - **Template:** contiene i template L^AT_EX comuni a tutti i documenti necessari per la loro compilazione. In particolare, in essi vengono definiti gli stili delle pagine, il frontespizio e i vari comandi creati per la stesura;

- **Verbali:** contiene i verbali redatti dal gruppo durante lo svolgimento del progetto. Per ogni documento verrà creata una sottocartella apposita, contenente i file necessari.
- **Informativi:** contiene dei files informativi; per esempio è presente una procedura per la creazione di un nuovo documento e un file contenente la lista di tutti i comandi \LaTeX utilizzati nei files di compilazione;
- **Pianificazione:** contiene tutti i grafici Gantt_G utilizzati nella pianificazione delle attività e nella gestione delle risorse;
- **Progettazione:** contiene tutti i diagrammi UML_G utilizzati durante la progettazione del software;
- **Scripts:** contiene gli scripts utilizzati per automatizzare alcune procedure; per esempio la compilazione dei documenti per una consegna.

4.2 Repository del codice

L'URL del repository_G del codice è: <https://github.com/nicolobissacco/7MonkeysCode>.
La struttura del repository_G verrà decisa durante la fase di *Progettazione di Dettaglio*.

5 Documenti

In questa sezione verranno presentati i vari standard adottati dal gruppo *Seven Monkeys* nella stesura dei documenti durante lo sviluppo del software.

5.1 Versionamento

Ogni documento deve specificare la propria versione, la quale sarà della seguente forma:

$$X.Y.Z$$

dove:

- **X**: numero che identifica la versione di rilascio. Ogni incremento causa l'azzeramento dei valori Y e Z. In particolare, questo parametro verrà incrementato ogni qual volta si consegneranno i documenti per una revisione;
- **Y**: numero che indica l'avvenuta verifica del documento. Nel momento in cui il verificatore ha controllato il documento, sarà tenuto ad incrementare questo parametro. Ogni incremento inoltre, comporta l'azzeramento del parametro Z;
- **Z**: numero che identifica una semplice modifica del documento. Ogni aggiunta o correzione al documento, comporta un incremento del parametro Z.

5.2 Template

Per facilitare la stesura e la manutenzione dei vari documenti, i file di compilazione sono stati strutturati in maniera tale che la composizione generale sia gestita da template validi per ogni documento. In particolare sono stati creati template per il frontespizio, per il layout, per i comandi utilizzati nei file e per la tabella delle modifiche. Ogni documento avrà una cartella interna denominata *Content*, in cui verranno collocati i file corrispondenti alle varie sezioni che lo compongono. La procedura che descrive come creare un nuovo documento è descritta in sez.6.1

5.3 Struttura dei documenti

5.3.1 Prima pagina

Ogni documento deve avere una prima pagina contenete le seguenti informazioni:

- Nome del progetto;
- Logo del gruppo;
- Nome del gruppo;
- E-mail esterna del gruppo;
- Nome del documento;
- La versione del documento⁶;
- Data di redazione del documento;
- Cognome e nome dei redattori del documento;
- Cognome e nome dei verificatori del documento;
- Cognome e nome di chi ha approvato il documento;

⁶Espressa come indicato in sezione 5.1.

- Tipo d'uso del documento⁷;
- La lista di distribuzione del documento;
- Un sommario, contenente una breve descrizione del documento.

5.3.2 Diario delle modifiche

Nella seconda pagina di ogni documento deve essere presente il diario delle modifiche. Ogni riga del diario delle modifiche deve contenere le seguenti informazioni:

- Una breve descrizione sulle modifiche effettuate;
- Il cognome e nome di chi ha effettuato la modifica;
- Il ruolo dell'autore della modifica;
- La data in cui è stato modificato il documento⁸;
- La versione del documento dopo la modifica.

5.3.3 Indici

In ogni documento, tranne che nel *Glossario*, deve essere presente un indice delle sezioni. Nel caso in cui il documento contenga immagini e/o tabelle, devono essere presenti anche i relativi indici.

5.3.4 Formattazione di una pagina

L'intestazione di ogni pagina deve avere le seguenti informazioni:

- Il logo del gruppo;
- Nome del gruppo;
- La sezione corrente all'interno del documento.

A piè di pagina deve invece esserci:

- Nome del documento;
- La versione corrente del documento;
- Numero di pagina nel formato “**Pagina X di N**”, dove “X” è il numero della pagina corrente e “N” è il numero di pagine totali del documento.

5.4 Classificazione dei documenti

5.4.1 Documenti informali

Si definiscono documenti informali tutti quei documenti che sono ancora in fase di stesura e devono ancora essere approvati dal *Responsabile di Progetto*.

Tali documenti sono ad *esclusivo* uso interno e non potranno essere divulgati a terze parti, prima di essere stati verificati ed approvati.

Una volta approvati, i documenti diventeranno formali e pronti per la distribuzione verso l'esterno.

Tutti i documenti informali dovranno essere salvati nel repository^G nell'apposita cartella⁹. Tali documenti devono essere rinominati osservando le seguenti regole:

⁷Potrà essere interno o esterno.

⁸La data deve essere espressa come riportato in sezione 5.5.4.

⁹Per maggiori dettagli vedere la sezione 4.1.

- La prima lettera di ogni parola, che non sia una preposizione, deve essere necessariamente maiuscola;
- Gli spazi devono essere sostituiti con il carattere “_” (underscore);
- I caratteri accentati devono essere sostituiti con i medesimi caratteri, ma privi di accento.

Un esempio di nome è il seguente: *Studio_di_Fattibilita.pdf*.

5.4.2 Documenti formali

Si definiscono documenti formali tutti i documenti che sono stati approvati dal *Responsabile di Progetto*¹⁰ e sono pronti per essere rilasciati.

Quando un documento diventa formale potrà essere visionato da terze parti.

Il file pronto per il rilascio dovrà seguire le stesse norme di nomenclatura del file in fase di stesura con l’aggiunta, alla fine del nome, della versione separata dal carattere “-” (trattino) e preceduta da una “v” (es. *Studio_di_Fattibilita-v1.0.0.pdf*).

5.4.3 Verbalì

Per verbalì si intendono tutti quei documenti redatti come promemoria in seguito ad un incontro.

Tutti i verbalì esterni saranno redatti dal *Responsabile di Progetto*, mentre i verbalì interni saranno redatti da un qualsiasi membro presente all’incontro.

I verbalì dovranno essere denominati secondo il seguente criterio:

Verbale{numero del verbale}_ {tipo di incontro}_ {data incontro}

dove:

- **Numero del verbale:** indica un numero progressivo che identifica il verbale¹¹;
- **Tipo di incontro:** indica il tipo di incontro avvenuto, che potrà essere:
 - **Interno:** nel caso si tratti di un incontro interno;
 - **Esterno:** nel caso si tratti di un incontro esterno.
- **Data incontro:** indica la data in cui è avvenuto l’incontro¹².

La prima pagina di ogni verbale deve contenere le seguenti informazioni:

- **Data**¹³;
- **Luogo ritrovo:** sintetica descrizione del luogo, correlata da indirizzo e città;
- **Ora inizio-fine:** espresso nel seguente formato ora inizio-fine: hh:mm-hh:mm;
- **Membri assenti:** i membri assenti alla riunione.

5.5 Norme tipografiche

Al fine di evitare incoerenza tra le diverse parti dei documenti, si rimanda a questa sotto-sezione per tutte le informazioni riguardanti l’ortografia, la tipografia e l’assunzione di uno stile uniforme in tutti i documenti.

¹⁰Fatta eccezione per il *Piano di Progetto*, che viene approvato da un qualsiasi altro membro del gruppo.

¹¹Tale numero dovrà partire da uno.

¹²La data dovrà essere formattata come indicato in sezione 5.5.4.

¹³Formato espresso come 5.5.4

5.5.1 Stile di testo

- **Grassetto:** il grassetto si deve utilizzare nei seguenti casi:
 - **Elenchi puntati:** per evidenziare l’oggetto trattato nel paragrafo;
 - **Altri casi:** è inoltre possibile utilizzare il grassetto per evidenziare termini particolarmente rilevanti.
- **Corsivo:** il corsivo deve essere utilizzato nei seguenti casi:
 - **Ruoli:** ogni riferimento a un ruolo deve essere scritto in corsivo (esempio: *Responsabile di Progetto*);
 - **Citazioni:** ogni citazione ad una fonte esterna va fatta tramite l’uso del corsivo;
 - **Documenti:** ogni riferimento a un documento deve essere scritto in corsivo (esempio: *Glossario*);
 - **File e directory:** ogni trascrizione del nome di un file o di una directory, deve essere fatta usando il corsivo;
 - **Altri casi:** in altri casi può essere necessario usare il corsivo, come per evidenziare termini particolarmente significativi.
- **Maiuscolo:** l’uso del maiuscolo è strettamente limitato alla trascrizione di acronimi;
- **L^AT_EX:** ogni riferimento a L^AT_EX va fatto tramite il comando `\LaTeX`.

5.5.2 Punteggiatura

- **Punteggiatura:** qualsiasi segno di punteggiatura non può seguire un carattere di spazio;
- **Punti ellittici:** i punti di sospensione devono essere inseriti esclusivamente tramite il comando `LATEX \dots`, immediatamente dopo l’ultimo carattere non di spaziatura;
- **Parentesi:** il periodo racchiuso tra le parentesi non deve mai iniziare con un carattere di spazio e non deve mai terminare con un carattere di spazio o punteggiatura;
- **Lettere maiuscole:** le lettere maiuscole vanno poste dopo il punto, il punto esclamativo, il punto interrogativo e all’inizio di ogni elemento di un elenco puntato. Inoltre viene utilizzata la lettera maiuscola per i ruoli di progetto, i nomi dei documenti, le fasi di progetto, le revisioni di progetto, oltre che dove imposto dalla lingua italiana.

5.5.3 Composizione del testo

- **Elenchi:** la prima parola che segue l’oggetto di indentazione deve avere la prima lettera minuscola. Ogni punto dell’elenco deve terminare con un carattere di punto e virgola, tranne l’ultimo che termina con un punto;
- **Note a piè di pagina:** ogni nota a piè di pagina deve iniziare con la prima lettera della prima parola maiuscola non preceduta da caratteri di spazio. Ogni nota a piè di pagina deve terminare con un punto;
- **Pedice “G”:** il pedice_G viene utilizzata in corrispondenza di termini tecnici e acronimi presenti nei documenti *Glossario*¹⁴.

¹⁴Le occorrenze dei termini presenti nei titoli, nelle note a piè di pagina e negli oggetti degli elenchi non andranno evidenziati.

5.5.4 Formati ricorrenti

- **Nomi propri:** l'utilizzo dei nomi propri deve seguire la forma "Cognome Nome";
- **Path:** per inserire indirizzi web o indirizzi mail, deve essere utilizzato *esclusivamente* il comando `\url`;
- **Date:** ogni data deve essere espressa seguendo lo standard ISO 8601¹⁵:

$$AAAA - MM - GG$$

dove:

- **AAAA:** rappresenta il formato dell'anno scritto con tutte e quattro le cifre;
- **MM:** rappresenta il mese scritto con esattamente due cifre¹⁶;
- **GG:** rappresenta il giorno scritto con esattamente due cifre.¹⁷
- **Riferimenti ai documenti:** ci si riferirà ai vari documenti scrivendoli in corsivo e mettendo una lettera maiuscola per ogni parola che non sia un articolo (ad esempio *Norme di Progetto*).
Nel caso ci sia la necessità di riferirsi ad una versione specifica del documento, essa andrà indicata (esempio: *Norme di Progetto v2.0.0*);
- **Nomi propri:** i nomi propri di persona devono essere scritti nel formato "Cognome Nome";
- **Nome del gruppo:** ci si riferirà al gruppo Seven Monkeys con la dicitura "*Seven Monkeys*", con il nome del gruppo in corsivo. Per la corretta scrittura è stato creato un comando `\authorName`;
- **Nome del proponente:** ci si riferirà al proponente con "Department of Information Engineering". Per la corretta scrittura è stato creato il comando `\proposerName`;
- **Nome del progetto:** ci si riferirà al progetto solo con "Romeo". Per la corretta scrittura è stato creato il comando `\project`;
- **Sigle:** le sigle andranno usate *esclusivamente* all'interno di tabelle o diagrammi secondo il seguente formalismo:
 - Re per indicare il *Responsabile di Progetto*;
 - Ad per indicare il *Amministratore di Progetto*;
 - Ve per indicare i *Verificatori*;
 - Pr per indicare i *Programmatori*;
 - Pj per indicare i *Progettisti*;
 - An per indicare gli *Analisti*.

5.6 Componenti grafiche

5.6.1 Immagini

Tutte le immagini utilizzate all'interno dei documenti dovranno avere formato `PDFG` o `PNGG`; è consigliato utilizzare `PDFG` vettoriale in quanto la sua risoluzione non dipende dal ridimensionamento dell'immagine.

La conversione di un immagine in formato `PDFG` è consentita grazie all'utilizzo del software

¹⁵http://it.wikipedia.org/wiki/ISO_8601

¹⁶Nel caso lo si possa scrivere utilizzando solo una cifra, si impone di anteporre uno zero al numero.

¹⁷Vedi la nota precedente.

Gimp¹⁸.

Le immagini devono essere accompagnate da una didascalia che inizia con la parola “Figura” con la prima lettera maiuscola, seguita dal numero della figura, dal carattere “due punti” e da una breve descrizione della figura¹⁹.

5.6.2 Tabelle

Ogni tabella deve essere utilizzata all’interno dei documenti con lo scopo di proporre informazioni in modo ordinato e coerente. Le tabelle dovranno essere accompagnate da una didascalia che inizia con la parola “Tabella” con la prima lettera maiuscola, seguita dal numero della tabella, dal carattere “due punti” e da una breve descrizione non banale della tabella²⁰.

Nel caso in cui il valore di una cella sia zero e il dato non è significativo, esso verrà omesso.

5.7 Glossario

Il *Glossario* contiene, in ordine lessicografico, la descrizione dei vari termini utilizzati all’interno degli altri documenti, che necessitano di un approfondimento in quanto possono generare ambiguità. Ogni occorrenza di un termine presente nel *Glossario*, è marcato nei documenti con una “**G**” in pedice.

È preferibile inserire un termine privo di definizione, piuttosto che rimandare l’inserimento dello stesso nel glossario.

¹⁸<http://www.gimp.org>

¹⁹Con breve si vuole intendere che la descrizione deve stare in una riga.

²⁰Vedi nota precedente.

6 Procedure e regole

Nella seguente sezione sono riportate l'elenco di procedure e regole seguite dal gruppo *Seven Monkeys*.

6.1 Creazione di un documento

La creazione di un nuovo documento è a discrezione esclusiva del *Responsabile di Progetto*. Nel momento in cui sia necessario creare un nuovo documento, egli dovrà seguire la seguente procedura (fig. 2):

1. Creare una nuova cartella all'interno di Documenti/ e denominarla con lo stesso nome del documento che si vuole creare;
2. Copiare la cartella *doc_to_modify* (contenuta nella cartella *Template/*) ed il suo contenuto, all'interno della cartella creata al punto 1;
3. Creare una cartella *Content* all'interno della cartella creata al punto 1. Questa directory conterrà i file di compilazione corrispondenti alle varie sezioni del documento;
4. Rinominare il file *template.tex* presente nella cartella *doc_to_modify*, utilizzando lo stesso nome del documento che si vuole creare. Successivamente, spostarlo all'interno della cartella creata al punto 1;
5. Compilare i campi dei file template precedentemente copiati, in base al documento che si vuole creare;
6. Creare i vari file di compilazione corrispondenti alle sezioni del documento all'interno della cartella *Content*.

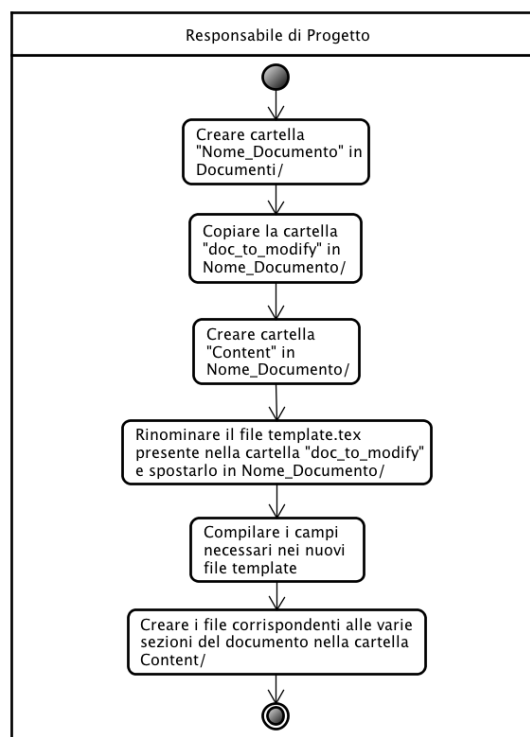


Figura 2: Procedura di creazione di un documento

6.2 Avanzamento di un documento

Ogni qualvolta un redattore ritiene che la stesura di un documento sia terminata andrà seguita la seguente procedura (fig. 3):

1. Il redattore dovrà contattare il *Responsabile di Progetto* per informarlo della terminazione della redazione del documento;
2. Il *Responsabile di Progetto* provvederà ad assegnare la verifica del documento ad uno dei *Verificatori* disponibili che non sia in conflitto d'interessi;
3. Il *Verificatore* provvederà alla verifica del documento;
4. Se sono stati trovati errori, il *Verificatore* provvederà alla creazione di un ticket di correzione. Successivamente il redattore correggerà gli errori trovati e infine si torna al passo uno;
5. Altrimenti il documento verrà approvato dal *Responsabile di Progetto*.

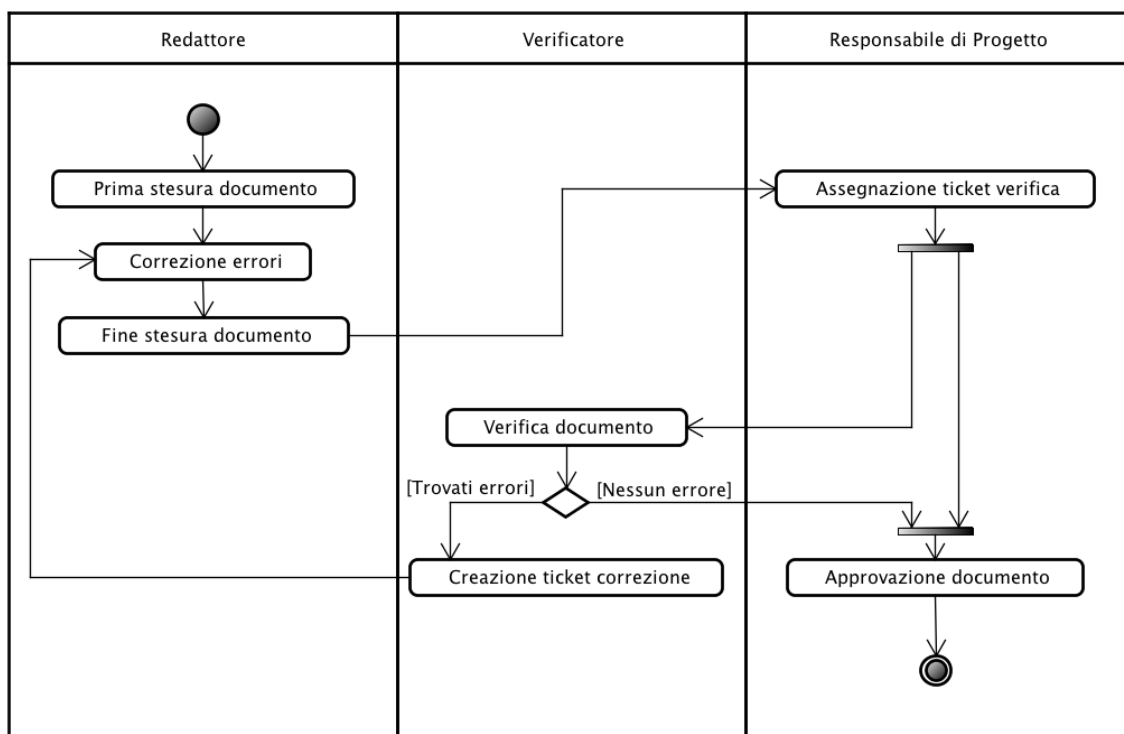


Figura 3: Procedura di avanzamento di un documento

6.3 Interazione con il repository

Per evitare conflitti nel repository, è *indispensabile* eseguire le seguenti operazioni di sincronizzazione all'inizio e alla fine di ogni sessione di lavoro (fig. 4):

1. **Pull:** prima di iniziare la sessione di lavoro, per ottenere i file aggiornati;
2. **Commit:** al termine di una sessione di lavoro, per commentare le modifiche effettuate;
3. **Pull:** subito prima di eseguire l'operazione di push;
4. **Push:** subito dopo l'operazione di pull, per caricare le modifiche nel repository remoto;
5. **Merge:** se durante l'operazione di push sorgono dei conflitti a seguito di modifiche effettuate da un altro membro.

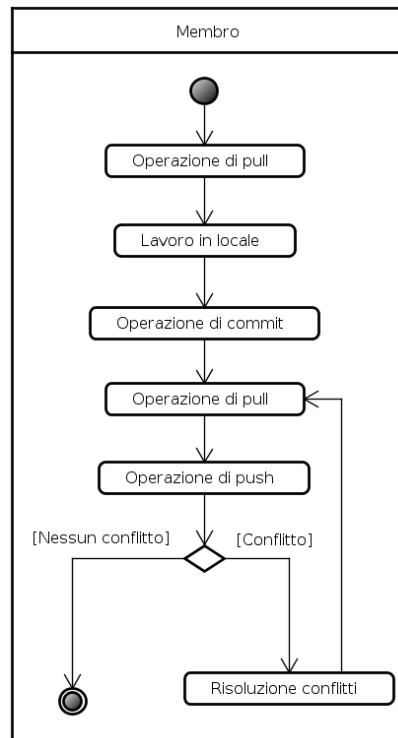


Figura 4: Procedura per l'utilizzo del Repository

I commenti aggiunti al commit dopo la modifica di un file devono essere chiari e devono specificare le modifiche effettuate. Per garantire una buona leggibilità dei commenti, i commit devono essere effettuati per ogni file modificato o gruppo di files dello stesso contesto. Inoltre, per evitare conflitti, è *sconsigliato* lavorare contemporaneamente sugli stessi file.

6.4 Gestione del Glossario

6.4.1 Inserimento di un termine

Per l'inserimento di un termine nel *Glossario* si dovrà seguire la seguente procedura (fig. 5):

1. Prima dell'inserimento di un termine andrà contattato il *Responsabile di Progetto* per proporre l'inserimento del termine;
2. Solo nel caso in cui il *Responsabile di Progetto* approvi l'inserimento del termine, esso andrà inserito nel *Glossario* nell'ordine lessicografico corretto;
3. Successivamente all'inserimento, si dovrà provvedere a evidenziare ogni occorrenza del termine nei vari documenti.

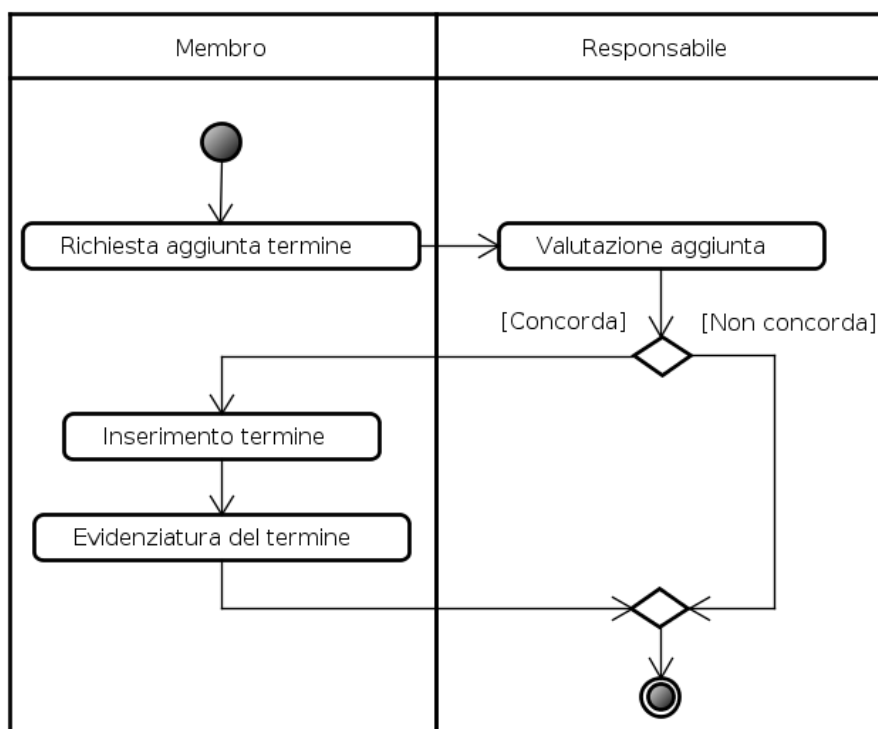


Figura 5: Procedimento per l’inserimento di un termine.

6.4.2 Eliminazione di un termine

Ogni membro del gruppo per richiedere l’eliminazione di un termine del *Glossario* dovrà seguire la seguente procedura (fig. 6):

1. Dovrà essere contattato il *Responsabile di Progetto* per proporre l’eliminazione;
2. Nel caso in cui il *Responsabile di Progetto* approvi, il termine andrà cancellato dal *Glossario*;
3. Successivamente si provvederà a togliere tutte le marcature del termine, all’interno di tutti i documenti.

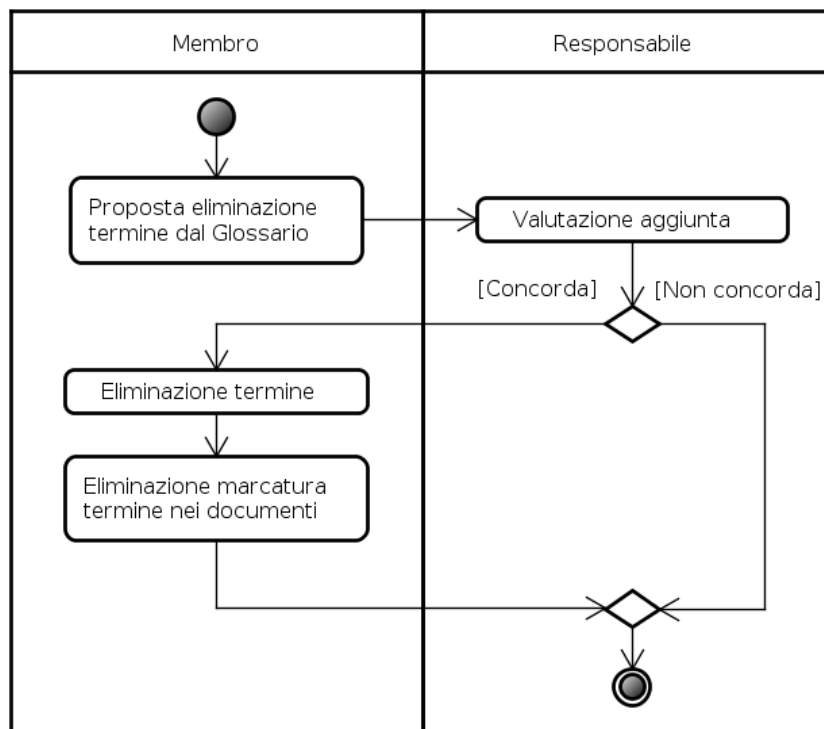


Figura 6: Procedimento per l'eliminazione di un termine

6.5 Rotazione dei ruoli

Durante lo svolgimento del progetto Romeo, è necessario che tutti i componenti del gruppo ricoprano tutti i vari ruoli definiti nel *Piano di Progetto v2.0.0*. Nel precedente documento verranno inoltre specificate le assegnazioni dei ruoli alle risorse, ripartite temporalmente e il quantitativo di ore che ciascuna risorsa dovrà svolgere in veste di un determinato ruolo. Per pianificare l'impiego delle risorse in tale senso, il *Responsabile di Progetto* si ispirerà alle seguenti regole:

- Ogni risorsa dovrà ricoprire tutti i ruoli;
- Il carico di ore individuali dovrà essere equo.

Riportiamo di seguito le direttive che regolano l'assegnazione e la modalità di rotazione dei ruoli.

I ruoli verranno assegnati alle risorse in modo tale che cambino al termine di una macro-fase (vedi *Piano di Progetto v2.0.0*). Dato che i ruoli da ricoprire sono in quantità minore rispetto alle risorse, sarà necessario che una o più risorse ricoprano più ruoli durante la stessa macro-fase. Per regolamentare questa ulteriore rotazione, le risorse ricopriranno un singolo ruolo per settimana, ruotando tra i ruoli assegnati nella specifica macro-fase.

Ogni componente del gruppo potrà consultare, in qualsiasi momento, i diagrammi di Gantt_G che descrivono la gestione delle risorse e dei ruoli, in maniera tale che ognuno potrà sempre essere consapevole del ruolo ricoperto dagli altri componenti.

7 Processi

7.1 Gestione di progetto

Sarà il *Responsabile di Progetto* ad avere l'onere di gestire il progetto durante tutto il suo ciclo di vita.

Il *Responsabile di Progetto* avrà il compito di:

- Pianificare le varie attività;
- Gestire le risorse;
- Analizzare e prevenire i rischi.

7.1.1 Pianificare attività

Per pianificare le varie attività da svolgere il *Responsabile di Progetto* dovrà utilizzare GanttProject_G per realizzare i vari diagrammi di Gantt_G.

7.1.2 Gestione delle risorse

Per gestire le varie risorse disponibili durante lo svolgimento del progetto, il *Responsabile di Progetto* dovrà creare dei diagrammi delle risorse, per pianificare la quantità di ore che ogni risorsa dovrà dedicare a ciascuna attività. ogni risorsa disponibile quante ore di ciascuna attività.

Per disegnare i diagrammi di attività, verrà impiegato GanttProject_G, un applicativo open source multiplatforma. Inoltre, per assegnare i compiti alle risorse disponibili, il *Responsabile di Progetto* utilizzerà il sistema di ticketing offerto da GitHub_G²¹.

7.1.3 Analisi e prevenzione dei rischi

Durante l'intero periodo di svolgimento del progetto, il *Responsabile di Progetto* dovrà analizzare i rischi che possono incombere. Inoltre dovrà cercare di prevenirli e trovare delle contromisure ad essi.

7.2 Analisi dei Requisiti

7.2.1 Studio di Fattibilità

In seguito alla pubblicazione dei capitolati d'appalto, il *Responsabile di Progetto* avrà il compito di convocare delle riunioni per discutere sugli aspetti positivi e negativi insiti in essi. Sarà compito degli *Analisti* redigere lo *Studio di Fattibilità* in base a quanto emerso durante le suddette riunioni.

Nello *Studio di Fattibilità*, per ogni capitolato, dovrà essere indicato:

- **Descrizione:** verrà fornita una breve descrizione del progetto proposto;
- **Dominio tecnologico ed applicativo:** saranno indicate le tecnologie richieste per la realizzazione del progetto e una breve descrizione del contesto applicativo in cui si colloca il prodotto richiesto nel capitolato;
- **Valutazione complessiva:** verranno espone le considerazioni emerse durante le discussioni. Infine si indicheranno le motivazioni principali per cui si è deciso di scartare o adottare il capitolato.

²¹Per maggiori informazioni consultare la sezione 8.

7.2.2 Analisi dei Requisiti

Il documento *Analisi dei Requisiti* sarà redatto dagli *Analisti*. In esso saranno inseriti tutti i requisiti ed i casi d'uso_G emersi durante l'attività di analisi.

Requisiti

Ogni requisito dovrà essere il più completo e non ambiguo possibile. I requisiti dovranno essere classificati per tipo e priorità, utilizzando la seguente notazione:

R[importanza][tipo][codice]

dove:

- **Importanza:** può assumere i seguenti valori:
 - **0:** in questo caso si tratta di un requisito obbligatorio;
 - **1:** in questo caso si tratta di un requisito desiderabile;
 - **2:** in questo caso si tratta di un requisito opzionale.
- **Tipo:** può assumere i seguenti valori:
 - **F:** per indicare un requisito funzionale;
 - **Q:** per indicare un requisito di qualità;
 - **P:** per indicare un requisito prestazionale;
 - **V:** per indicare un requisito di vincolo.
- **Codice:** rappresenta il codice univoco di ogni requisito, il quale va indicato in forma gerarchica.

I requisiti verranno rappresentati nel documento *Analisi dei Requisiti*, in una tabella contenente l'identificativo, una breve descrizione e la fonte da cui sono stati tratti.

Casi d'uso

Per ogni caso d'uso_G dovrà essere fornito:

- **Identificativo:** ogni caso d'uso_G va tracciato con un codice. Quest'ultimo sarà composto dalle due lettere maiuscole "UC", seguite dal numero identificativo del caso d'uso_G (es. **UC2.4.6**);
- **Diagramma:** verrà utilizzato UML_G per creare i diagrammi dei caso d'uso_G, con lo scopo di semplificare la comprensione dei requisiti.
Di seguito sono riportate le convenzioni adottate nella realizzazione dei diagrammi:
 - **Standard:** verrà adottato lo standard UML_G 2.0;
 - **Lingua:** verrà adottata la lingua italiana all'interno dei diagrammi.
- **Descrizione:** una breve didascalia che descrive il caso d'uso_G;
- **Flusso principale:** verrà fornita una descrizione del flusso principale degli eventi del caso d'uso_G;
- **Scenario alternativo:** nel caso sia presente uno scenario alternativo, il caso d'uso_G dovrà essere correlato da una descrizione di tale scenario;
- **Attori:** gli attori coinvolti nel caso d'uso_G;
- **Precondizione:** la precondizione del requisito;
- **Postcondizione:** la postcondizione del requisito.

Tracciamento dei requisiti

Per il tracciamento dei requisiti si utilizzerà il software **ReqMonkeys**²², creato dal gruppo *Seven Monkeys*.

7.3 Progettazione

7.3.1 Specifica Tecnica

I *Progettisti* dovranno descrivere la progettazione ad alto livello dell'architettura del software e dei vari componenti nel documento di *Specifica Tecnica*. Si cercherà di creare un'architettura il più *estendibile e manutenibile* possibile in modo da facilitare future modifiche al software.

Diagrammi UML

Verrà utilizzato il linguaggio UML_G per definire i seguenti diagrammi:

- Diagrammi delle classi;
- Diagrammi dei package;
- Diagrammi delle attività;
- Diagrammi di sequenza.

Di seguito sono riportate le convenzioni utilizzate nella realizzazione dei diagrammi UML_G:

- **Standard:** lo standard UML_G utilizzato è il 2.0;
- **Lingua:** la lingua utilizzata all'interno dei diagrammi è l'inglese.

Design pattern

I *Progettisti* dovranno utilizzare i design pattern_G più adatti al contesto, in modo da rendere l'applicazione il più efficiente possibile. Ogni design pattern_G utilizzato, verrà accompagnato da una breve descrizione e da un diagramma che ne esemplifica il funzionamento.

Classi di verifica

Andranno realizzate delle classi di verifica, da utilizzare per testare che le varie componenti del sistema abbiano un comportamento corretto rispetto alle attese.

Stile di progettazione

Durante la Progettazione, una particolare attenzione sarà posta su:

- **Puntatori:** non andranno utilizzati puntatori se non *strettamente* necessario, in quanto rendono il codice di difficile comprensione;
- **Ricorsione:** non andrà utilizzata la ricorsione a meno che non sia necessaria, in caso andrà fornita una dimostrazione induttiva sulla correttezza del metodo in questione;
- **Annidamento di cicli:** all'interno di un metodo non dovranno esserci annidamenti di cicli con una profondità maggiore a cinque.

7.3.2 Definizione di Prodotto

Successivamente alla *Specifica Tecnica*, i *Progettisti* dovranno produrre la *Definizione di Prodotto* del software in questione. Tale documento andrà più in dettaglio nella varie componenti del software rispetto alla *Specifica Tecnica*.

²²Per maggiori informazioni vedere 9.5.

Diagrammi UML

Verrà utilizzato il linguaggio UML_G per definire i seguenti diagrammi:

- Diagrammi delle classi;
- Diagrammi delle attività;
- Diagrammi di sequenza.

Di seguito sono riportate le convenzioni utilizzate nella realizzazione dei diagrammi UML_G:

- **Standard:** lo standard UML_G utilizzato è il 2.0;
- **Lingua:** la utilizzata all'interno dei diagrammi è l'inglese.

Descrizione di classi

Ogni classe progettata dovrà essere descritta nella *Definizione di Prodotto*.

Per ogni classe andrà fornito:

- **Diagramma:** un diagramma UML_G che rappresenta la classe;
- **Descrizione:** una breve descrizione del funzionamento della classe in questione;
- **Utilizzo:** il contesto di utilizzo della classe all'interno del software;
- **Attributi:** verranno indicati i vari attributi della classe, corredati da una breve descrizione;
- **Metodi:** verranno indicati i metodi della classe, corredati da una breve descrizione dei vari parametri e del metodo.

Test d'unità

I *Progettisti* dovranno provvedere alla creazione di test d'unità per verificare il corretto funzionamento delle varie componenti del sistema.

7.4 Verifica

La verifica di 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. Si descrivono ora le modalità ordinate e puntuali di verifica di processi, documenti, attività e codice alle quali ci si riferirà in questo documento e alle quali i *Verificatori* dovranno attenersi.

7.4.1 Tecniche di analisi

Esistono due tecniche di analisi: l'*analisi statica* e l'*analisi dinamica*.

Analisi statica può essere di tipo walkthrough o inspection.

- **Walkthrough:** questa tecnica di analisi prevede una lettura critica del codice o del documento prodotto. Tale tecnica è molto dispendiosa in termini di risorse, visto che viene applicata all'intero documento, senza avere una precisa idea di quale sia il tipo di anomalia e di dove ricercarla. Essa è però necessaria nelle prime fasi del progetto, vista l'inesperienza da parte del gruppo nell'attuare un tipo di verifica più precisa e mirata. Dopo una prima fase di lettura ed identificazione degli errori, si procede alla discussione degli stessi, proponendo le modifiche da apportare per garantirne la correzione. Il passo finale consiste nell'applicare le modifiche proposte, redigendo un rapporto preciso che elenchi le modifiche effettuate. Una caratteristica di questo tipo di analisi è che richiede l'utilizzo di più risorse umane;

- **Inspection:** questa tecnica di analisi presuppone l'esperienza da parte del verificatore, nell'individuare gli errori e le anomalie più frequenti. A tal scopo è necessaria una *lista di controllo*, stilata in una precedente analisi di tipo *walkthrough*, nella quale vengono elencate le sezioni più critiche. Questo permette quindi una verifica più rapida che non necessità della lettura dell'intero documento o codice, oltre che richiedere meno risorse umane. Dopo aver terminato l'analisi, sarà necessario stilare un rapporto di verifica che tenga traccia del lavoro svolto e delle modifiche apportate.

7.4.2 Verifica dei documenti

Il processo di verifica dei documenti viene eseguito ogni qualvolta un documento abbia subito una modifica e debba essere approvato. È compito del *Responsabile di Progetto* affidare ai *Verificatori* la verifica di un documento. Per eseguire una corretta verifica di un documento va seguita la seguente prassi:

1. **Controllo tipografico:** tramite l'utilizzo di `TeXmaker` e di `AspellG` verranno trovati errori tipografici presenti nel documento;
2. **Controllo lessicale:** il *Verificatore* dovrà controllare che il documento non contenga degli errori lessicali attraverso un'attenta lettura delle parti da verificare: se non è ancora presente una lista di controllo, si effettuerà un'analisi di tipo *walkthrough*, altrimenti si procederà all'analisi *inspection*;
3. **Controllo di glossario:** il *Verificatore* dovrà controllare che ogni termine presente nel glossario sia evidenziato correttamente all'interno del documento;
4. **Controllo di contenuto:** il *Verificatore* dovrà controllare che il documento contenga tutti gli argomenti da trattare, che non manchi nulla, e che sia impaginato con una struttura adeguata;
5. **Rispetto delle norme di progetto:** il *Verificatore* dovrà controllare che il documento segua le norme di progetto stabilite;
6. **Stesura lista di controllo:** si dovrà stilare una lista degli errori più frequenti, in modo da facilitare le successive attività di verifica dei documenti;
7. **Calcolo dell'indice di Gulpease:** su ogni documento redatto il *Verificatore* dovrà calcolare l'indice di Gulpease. Nel caso in cui l'indice non rispettasse il range di valori specificato nel *Piano di Qualifica v2.0.0*, sarà necessario eseguire un *walkthrough* del documento alla ricerca delle frasi troppo lunghe o complesse;
8. **Segnalazione degli errori riscontrati:** una volta terminata la verifica il *Verificatore* dovrà creare i ticket opportuni per segnalare gli errori riscontrati, come spiegato in sezione 8.2.

7.4.3 Verifica dei diagrammi

Al *Verificatore* è richiesta la verifica dei seguenti diagrammi `UMLG`:

- **Diagrammi di flusso:** la verifica dei diagrammi di flusso deve avvenire manualmente controllando che essi aderiscano allo standard `UMLG 2.0` e rappresentino in modo corretto la procedura in questione;
- **Diagrammi dei casi d'uso:** la verifica dei diagrammi dei casi d'uso deve avvenire manualmente, controllando che essi rispettino lo standard `UMLG 2.0`. In particolare, deve essere verificato il corretto utilizzo di inclusioni ed estensioni; inoltre, si deve assicurare che ci sia la massima corrispondenza tra il diagramma e la descrizione testuale.

7.4.4 Verifica del codice

Nella verifica del codice verranno utilizzati i seguenti strumenti:

Analisi statica

Per la verifica statica in Jenkins sono stati integrati i seguenti strumenti:

- **Compilatori:** il compilatore è il primo strumento di analisi, in quanto segnala in modo inequivocabile errori e warning. Dato che il software dovrà essere multiplatforma, i sorgenti del progetto dovranno essere compilati con i seguenti compilatori:
 - **GCC**²³: compilatore per ambiente Linux_G;
 - **Clang**²⁴: compilatore per ambiente Mac OS_G;
 - **MinGW**²⁵: compilatore per ambiente Windows_G.
- **CppCheck**²⁶: si tratta di un analizzatore statico di codice per il C++_G, che permette la rilevazione di alcuni errori;
- **CCCC**²⁷ *C and C++ Code Counter*: Misura metriche riguardanti codice sorgente C++_G. Tra le più rilevanti ci sono: complessità ciclomatica, linee di codice per linee di commento, metodi per classe ed accoppiamento tra gli oggetti.

Analisi dinamica

L'analisi dinamica verrà effettuata con:

- **Valgrind**²⁸: tool_G che permette di rilevare memory leak_G, altri problemi di memoria comuni in C++_G ed effettuare il profiling_G del software. Compatibile solo con sistemi Unix;
- **Very Sleepy**: tool per il profiling_G del software in ambiente Windows_G;
- **Dr. Memory**²⁹: tool per il controllo della memoria in ambiente Windows_G;

Test d'unità

Per i test di unità sul codice si utilizzerà il framework_G Qt Test³⁰.

Per maggiori informazioni si rimanda al *Piano di Qualifica v2.0.0*.

Resoconto bug

Se durante la verifica del codice, il *Verificatore* trova dei bug_G, dovrà utilizzare lo strumento Mantis³¹ per segnalare i problemi ai *Programmatori*.

7.5 Codifica

Di seguito sono riportate le convenzioni che i *Programmatori* dovranno seguire durante la fase di Codifica.

Per rendere il codice più leggibile e facilmente comprensibile, i *programmatori* dovranno seguire le Qt_G Coding Convention e le Qt_G Styling Code. (Vedi sezione 1.3.1)

È consentita la possibilità di effettuare dei cambiamenti alle convenzioni stabilite, in seguito ad una decisione del *Responsabile di Progetto*.

²³<http://gcc.gnu.org>

²⁴<http://clang.llvm.org>

²⁵<http://www.mingw.org>

²⁶<http://cppcheck.sourceforge.net>

²⁷<http://cccc.sourceforge.net>

²⁸<http://valgrind.org>

²⁹<http://www.drmemory.org>

³⁰<http://qt-project.org/doc/qt-5.0/qttestlib/qtest-overview.html>

³¹<http://www.mantisbt.org>

7.5.1 Linguaggio di codifica

Una prima analisi del capitolato e dei suoi requisiti ha delineato come scelta ottimale l'uso del linguaggio C++ per lo sviluppo dell'applicativo richiesto.

7.5.2 Convenzioni di codifica

Di seguito è riportato l'insieme di norme e convenzioni che il gruppo *Seven Monkeys* seguirà nella scrittura e documentazione del codice.

L'unica lingua ammessa per i nomi di variabili, classi, metodi e commenti è l'inglese.

Nomenclatura

Per l'assegnazione di nomi a classi, variabili, metodi e costanti andranno seguite le seguenti regole:

- **Classi:** va utilizzata la notazione mixed case, con la prima lettera maiuscola;
- **Metodi:** va utilizzata la notazione mixed case, con la prima lettera minuscola;
- **Variabili:** va utilizzata la notazione mixed case, con la prima lettera minuscola;
- **Costanti:** va scritto il nome *interamente* in maiuscolo, separando le varie parole con il carattere “_” (underscore).

Intestazione di un file

Ad ogni file di codice dovrà corrispondere una singola classe propria.

Una classe dovrà essere suddivisa in un file *header* (.h) e in un file di *implementazione* (.cpp). Ogni file *header* dovrà necessariamente avere un'intestazione, la quale dovrà avere la seguente forma:

```

/!*
*\file Nome del file
*\author Cognome Nome (e-mail)
*\date Data di creazione del file
*\brief Descrizione breve della classe
*Descrizione dettagliata della classe
*/
/*
*Changes:
*+-----+-----+-----+-----+-----+
*| Version + Date + Programmer +          Changes          + Description |
*+-----+-----+-----+-----+-----+
*|
*|   x.y.z   | AAMMGG | Nome Cognome | ClassName::MethodName | Description   |
*|
*+-----+-----+-----+-----+-----+
*/

```

dove:

- **File:** deve essere il nome del file, comprendente di estensione (.h o .cpp);
- **Author:** deve essere il creatore del file e non necessariamente il programmatore che sta modificando il file attualmente;
- **Date:** deve essere la data di creazione del file, espressa come indicato in sezione 5.5.4;
- **Brief:** è una descrizione breve della classe (massimo due righe);
- **Changes:** rappresenta la tabella di avanzamento del file, comprensiva di modifiche, aggiunte ed eliminazioni di metodi della classe.
Nello specifico la tabella deve avere le seguenti righe:

- **Version:** la versione del file successivamente alla modifica;
- **Date:** la data in cui è stato modificato il file;
- **Programmer:** il programmatore che ha fatto la modifica, che può non essere necessariamente il creatore del file;
- **Changes:** rappresenta il metodo che ha subito cambiamenti;
- **Description:** una breve descrizione del cambiamento effettuato.

Commenti nei metodi

Sempre nel file *header* prima di ogni metodo, si dovrà inserire un commento che dovrà rispettare la seguente forma:

```
/*!brief Breve descrizione del metodo
*Descrizione dettagliata del metodo.
*\param Descrizione del primo parametro
*\param Descrizione dell'n-esimo parametro
*\return il tipo del valore di ritorno
*/
/*
*Pre-Condition: {La pre-condizione}
*Post-Condizione: {La post-condizione}
*/
```

dove:

- **Brief:** deve essere una breve descrizione del funzionamento del metodo (massimo due righe);
- **Pre-Condition:** deve essere la pre condizione che vale prima della chiamata del metodo;
- **Post-Condition:** deve essere la post condizione che varrà dopo la chiamata del metodo;
- **Return:** deve essere il tipo di ritorno;

Commenti nei file di implementazione

Nei vari file di implementazione è gradita la presenza di commenti in modo da facilitare la comprensione del codice. I commenti devono essere scritti nel modo più chiaro e descrittivo possibile. Ogni variabile di particolare importanza, o il quale utilizzo è particolarmente complesso, dovrà essere accompagnata da una breve descrizione del suo scopo. In particolare è gradita la presenza di *invarianti*, *pre* e *post condizioni* sui cicli, nel caso in cui essi abbiano almeno due livelli di annidamento.

8 Protocollo per la gestione del progetto

Il servizio GitHub_G, al suo interno incorpora un sistema per la gestione dei ticket e creazione di milestone_G. Si è quindi deciso di usufruire di tali servizi.

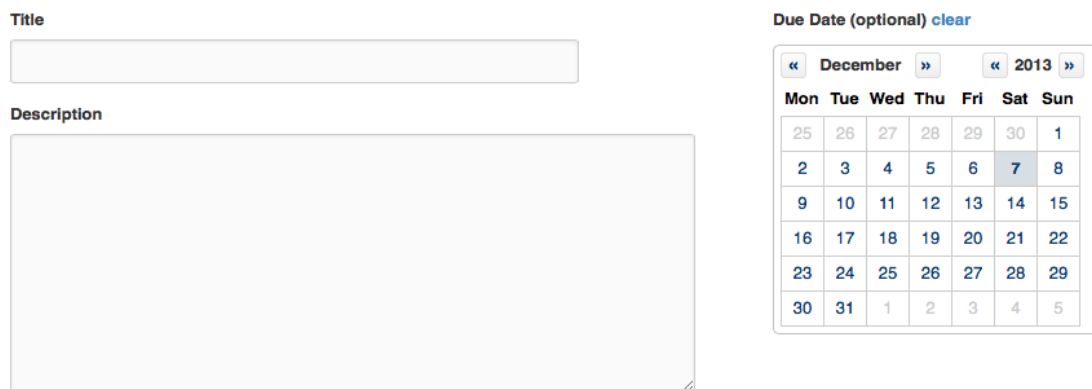
8.1 Creazione milestone

Il *Responsabile di Progetto* dovrà provvedere alla creazione di una milestone_G per la prossima revisione a cui il gruppo *Seven Monkeys* ha intenzione di partecipare.

Per creare una nuova milestone_G bisogna:

- Cliccare su issue;
- Andare su milestone_G;
- Premere su “Create a new milestone_G”;
- Infine vanno compilati i campi richiesti³².

Si potrà vedere lo stato di avanzamento della milestone_G guardando il numero di ticket completati rispetto ai ticket totali.



The screenshot shows the GitHub 'Create a new milestone' form. On the left, there is a 'Title' input field and a larger 'Description' text area. On the right, under the heading 'Due Date (optional) clear', there is a calendar widget for December 2013. The calendar shows days from 1 to 31, with the 7th of December highlighted in blue.

Figura 7: Interfaccia di GitHub per la creazione di una milestone

³²I possibili cambi da completare sono riportati in Figura 7.

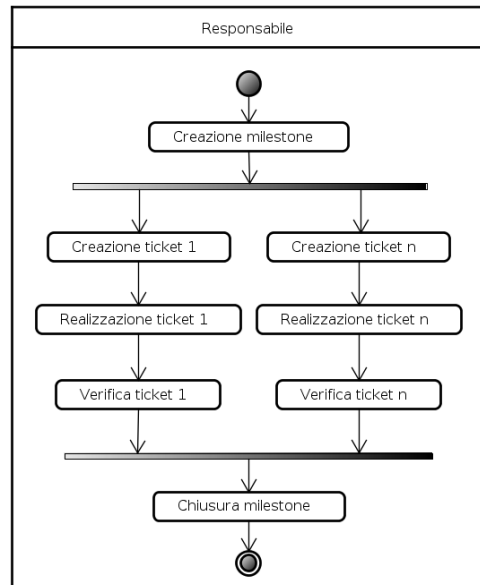


Figura 8: Modello per la gestione di una milestone

8.2 Creazione ticket

I ticket vengono creati da:

- **Responsabile di Progetto:** crea la maggioranza dei ticket;
- **Verificatore:** crea i ticket per segnalare le imprecisioni o errori trovati durante la verifica.

Ogni ticket dovrà essere assegnato ad un unico membro del team *Seven Monkeys* e dovrà avere le seguenti caratteristiche³³:

- **Destinatario:** a chi è rivolto l'attività indicata dal ticket;
- **Milestone:** la milestone_G a cui è associato il ticket;
- **File:** dovrà essere indicato il file oggetto dell'attività;
- **Label:** ogni ticket avrà una o più label associate; le label possibili sono le seguenti:
 - **Modifica:** ticket generalmente creato dal *Verificatore* per segnalare gli errori trovati;
 - **Verifica:** ticket creato dal *Responsabile di Progetto* per assegnare la verifica a uno dei *Verificatori*;
 - **Richiesta approvazione:** ticket creato da un *Verificatore* per richiedere l'approvazione di un documento;
 - **Creazione:** ticket creato dal *Responsabile di Progetto* per assegnare un compito a un generico membro del gruppo;
 - **Priorità alta:** label assegnata ad un ticket creato per la gestione di anomalie;
 - **Priorità media:** label assegnata ad un ticket creato per la gestione di discrepanze di media gravità;
 - **Priorità bassa:** label assegnata ad un ticket creato per la gestione di discrepanze non gravi.

³³In figura 3 è mostrata l'interfaccia per la creazione di ticket.

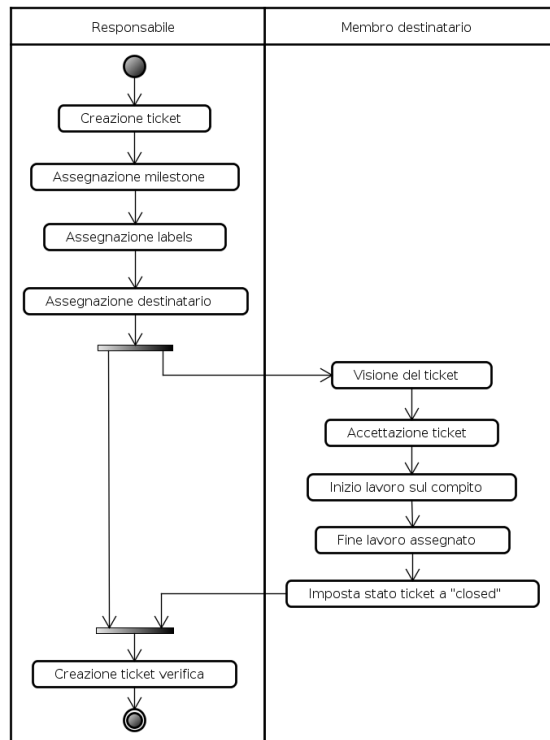


Figura 9: Modello di ticket per la creazione di un compito generico

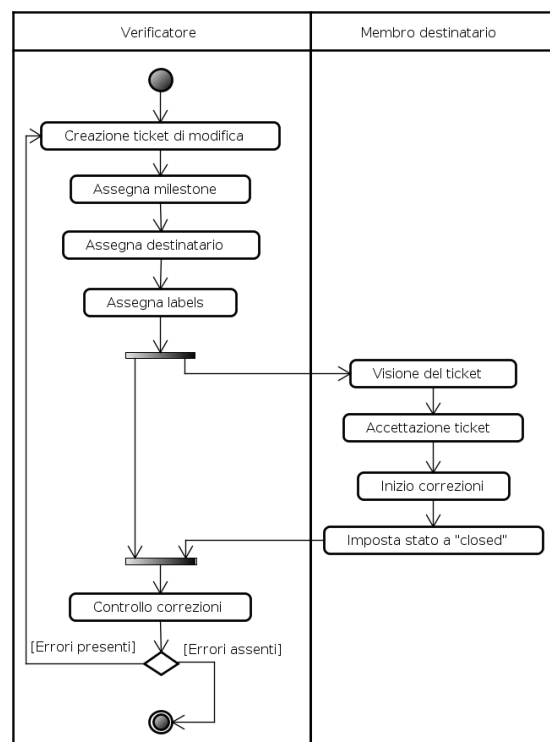


Figura 10: Modello di ticket per la creazione di un ticket di verifica

8.3 Esecuzione dei compiti

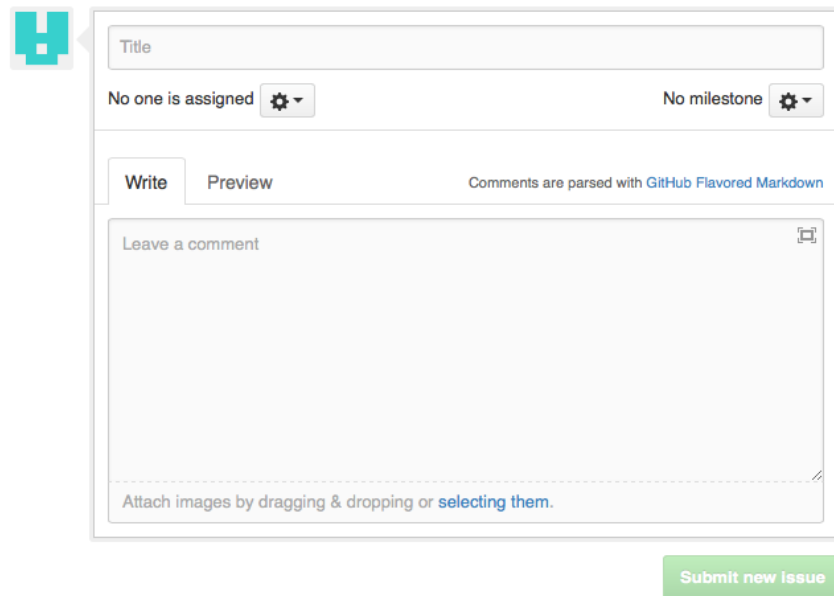


Figura 11: Interfaccia di GitHub per la creazione di un ticket.

Ogni membro del gruppo dovrà visionare i ticket a lui assegnati e, successivamente all'esecuzione del compito, dovrà cambiare lo stato del ticket in "closed". Quando il destinatario del ticket ne prenderà visione, dovrà confermarlo con un commento nella pagina web del ticket. Se il *Responsabile di Progetto* si rende conto che un ticket non è stato eseguito come atteso, "riaprirà" il ticket.

8.4 Chiusura della milestone

Una volta raggiunta la scadenza o terminati tutti i ticket, la milestone_G verrà chiusa. Successivamente il *Responsabile di Progetto* provvederà alla creazione di una nuova milestone_G e tutto il protocollo ricomincerà da capo.

9 Ambiente di Lavoro

In questa sezione verrà illustrato in dettaglio l'ambiente di lavoro che il team *Seven Monkeys* utilizzerà durante lo sviluppo del progetto Romeo.

9.1 Sistema operativo

Il sistema operativo utilizzato per lo sviluppo del progetto è a discrezione di ogni singolo componente del gruppo. Questa scelta è dovuta principalmente al fatto che il progetto dovrà supportare più piattaforme.

In particolare i vari membri del gruppo utilizzeranno i seguenti sistemi operativi:

- Ubuntu 12.04 64bit;
- Mac OS_G 10.9 64bit;
- Windows_G 8 64bit.

9.2 Coordinamento

Il coordinamento del gruppo avviene tramite:

- Repository_G Git_G;
- Dropbox_G;
- Google Drive_G;
- Google Calendar_G;
- Jenkins.

9.2.1 Repository Git

Sono stati presi in considerazione vari repository_G (Git_G, Mercurial, SVN) prima di scegliere di utilizzare Git_G.

La scelta è ricaduta su Git_G per i seguenti motivi:

- La possibilità di poter lavorare localmente, senza bisogno di essere connessi a internet;
- Git_G era già stato usato da vari membri del team;
- La possibilità di ignorare certe estensioni dei file (gitignore);
- La presenza di innumerevoli client, per chi non volesse utilizzarlo da shell.

Per i membri del gruppo, che non vogliono utilizzare la shell, sono consigliati i seguenti client:

- **Sourcetree**: disponibile per i sistemi Windows_G e Mac OS_G;
- **Giteye**: disponibile per i sistemi Linux_G.

9.2.2 Dropbox

In questo servizio cloud verranno messi tutti i file che non sono soggetti ad un controllo di versione.

Principalmente verrà utilizzato dai membri del gruppo per scambiarsi libri di testo, guide, ecc....

La presenza del client ufficiale disponibile per la maggior parte dei sistemi operativi ha fatto cadere la scelta su questo servizio cloud.

9.2.3 Google Drive

In quest'altro servizio cloud invece andranno messi tutti i documenti che:

- Non necessitano di alcun controllo di versione;
- Necessitano di una forte interattività tra i vari membri del gruppo;
- Sono accessibili direttamente da browser.

Principalmente Google Drive_G viene utilizzato per lavorare su dei file condivisi su Google Docs_G.

Un vantaggio di questo servizio è quello che i vari membri possono lavorare contemporaneamente sugli stessi documenti, tramite l'utilizzo di un browser.

9.2.4 Google Calendar

Google Calendar_G viene utilizzato dai membri del gruppo per la gestione delle risorse umane. È stato creato un calendario condiviso tra i vari membri del gruppo, in modo da notificare:

- In quali date un certo membro non è disponibile;
- In quali date c'è una riunione o un evento rilevante ai fini del gruppo.

Grazie alla gestione delle notifiche di Google Calendar_G, è possibile far in modo che 24 ore prima di un evento rilevante venga inviata una email a tutti i membri del gruppo come promemoria.

9.2.5 Jenkins

Per svolgere l'*integrazione continua* durante lo svolgimento del progetto si è deciso di appoggiarsi al servizio Jenkins³⁴.

Si è scelto di utilizzare tale software per il gran numero di plugin.

9.3 Pianificazione delle attività

Per pianificare la gestione di progetto e gestire le risorse umane si è scelto di utilizzare GanttProject_G.

La scelta è ricaduta su tale applicativo per i seguenti motivi:

- Portabilità, essendo il software scritto in Java;
- Open source;
- Permette la generazione di diagrammi di Gantt_G e delle risorse;
- Consente di esportare i diagrammi in formato PNG o HTML.

9.4 Tracciamento bug

Per il tracciamento dei bug_G è stato scelto di utilizzare l'applicativo web Mantis³⁵.

Tale software è disponibile all'indirizzo: <http://sevenmonkeys.altervista.org/mantisbt/index.php>.

Mantis è accessibile a tutti i membri del gruppo *Seven Monkeys* tramite login e password.

³⁴<http://jenkins-ci.org>

³⁵<http://www.mantisbt.org>



Figura 12: Schermata di login di mantis

9.5 ReqMonkeys

Per facilitare il tracciamento dei requisiti è stato creato dal *Amministratore di Progetto* il software **ReqMonkeys**, il quale è ospitato nel sito del gruppo all'indirizzo sevenmonkeys.altervista.org.

Il software permette di gestire il tracciamento e gestire requisiti, casi d'uso, fonti, test d'unità velocizzando le varie operazioni.

Nella figura sottostante è possibile vedere l'interfaccia della pagina principale di **ReqMonkeys**.

9.5.1 Analisi dei Requisiti

ReqMonkeys permette di tracciare i requisiti.

Infine il software permette di scaricare il file `LATEX` `Requisiti.tex` contenente le tabelle.

9.6 Strumenti per i documenti

9.6.1 L^AT_EX

Per la scrittura dei documenti è stato scelto di utilizzare il linguaggio di markup `LATEX`³⁶. Come editor è consigliato l'utilizzo di `TEXmaker`³⁷, il quale è disponibile per tutti i principali sistemi operativi.

9.6.2 Script

Per semplificare alcune azioni l'*Amministratore di Progetto* ha creato alcuni script.

Per facilitare la compilazione e il controllo dei documenti è stato creato un Makefile, disponibile nel repository_G (`/Scripts/Makefile`).

Il Makefile consente di:

- **Generazione di tutti i PDF:** tramite il comando `make RP` verranno generati tutti i PDF_G e salvati sulla cartella `Consegne/RR`;
- **Eseguire il controllo ortografico:** tramite il comando `make aspell` verrà invocato l'applicativo `AspellG` su tutti i documenti;
- **Eliminare i file generati:** tramite il comando `make clean` verranno eliminati tutti i file generati da vecchie compilazioni.

³⁶<http://latex-project.org/ftp.html>

³⁷<http://www.xmlmath.net/texmaker/>

9.6.3 Controllo ortografico

Per la verifica ortografica dei documenti scritti in L^AT_EX verrà utilizzato l'applicativo Aspell_G. La scelta è ricaduta su di esso, piuttosto che sul correttore ortografico integrato in T_EXmaker, perché consente di aggiungere termini al proprio dizionario.

L'uso di Aspell_G avverrà direttamente da shell tramite l'uso del comando:

```
aspell --mode=tex --lang=it check nomedocumento.tex
```

9.6.4 Diagrammi UML

Per quanto riguarda la modellazione dei diagrammi dei casi d'uso, diagrammi di attività e diagrammi di sequenza, si è deciso di adottare il software *Astah_G - Professional Edition*. Quest'ultimo è uno strumento multiplatforma che permette di disegnare svariate tipologie di grafici secondo lo standard UML2.0_G.

Per la realizzazione dei diagrammi delle classi e dei package_G verrà utilizzato il software Visual Paradigm³⁸. Tale software offre i seguenti vantaggi:

- Distinzione tra i diagrammi delle classi e package_G;
- Il software è multiplatforma;
- Generazione automatica del codice dello “scheletro” delle classi;
- Facilità di utilizzo del software.

9.7 Strumenti per lo sviluppo

L'*Amministratore di Progetto* ha configurato una macchina virtuale Windows_G utilizzando il software **VirtualBox**³⁹.

La macchina virtuale contiene tutto il software di lavoro utilizzato dal gruppo *Seven Monkeys*.

Come sistema operativo è stato scelto Windows_G in quanto lo sviluppo principale di Romeo deve avvenire per tale sistema.

9.7.1 Framework

Per la realizzazione della grafica del progetto è stato scelto di utilizzare il framework_G Qt_G. I motivi che hanno portato a tale scelta sono:

- **Qt Linguistic:** è disponibile un tool_G, che permette il supporto a varie lingue;
- **Qt Designer:** è presente questo tool_G, il quale facilita la creazione di interfacce;
- **Esperienza:** tutti i membri del team hanno già utilizzato il framework_G Qt_G in precedenza nel corso di Programmazione ad Oggetti.

La versione utilizzata delle librerie Qt_G è la 5.1.

9.7.2 Librerie

Per la realizzazione di alcune parti fondamentali del progetto, verranno utilizzate delle librerie esterne al framework_G sopra descritto, come suggerito dai proponenti. In particolare verranno impiegate le librerie ITK_G (v4.5.0) e VTK_G (v6.1.0) per la manipolazione dei vari formati d'immagine che il software dovrà saper gestire. Queste librerie offrono la possibilità di importare ed esportare le immagini all'interno del software, e di poter quindi applicarne gli algoritmi che effettuano i calcoli voluti.

³⁸<http://www.visual-paradigm.com>

³⁹<https://www.virtualbox.org>

9.7.3 Ambiente di codifica

Per la scrittura del codice è stato scelto di utilizzare l'IDE `QtG Creator`.

È stato scelto poiché, oltre ad integrarsi direttamente con le librerie `QtG`, fornisce degli strumenti di editing, debugging e la possibilità di utilizzare la documentazione ufficiale anche offline.

Inoltre `QtG Creator` si integra con `GitG`.

La versione di `QtG Creator` utilizzata è la 2.8.1.

9.7.4 Documentazione

Per la documentazione del codice è stato scelto di usare il sistema Doxygen⁴⁰, il quale permette una generazione automatica della documentazione a partire dal codice.

Il sistema estrae la documentazione dai commenti inseriti nel codice sorgente e dalla dichiarazione delle strutture dati.

Doxygen permette di generare la documentazione in formato HTML o `LATEX`.

⁴⁰<http://www.stack.nl/~dimitri/doxygen/>

A Lista di controllo

Nella verifica dei documenti, si sono rilevati, più frequentemente, errori dovuti a:

- **Non aderenza delle Norme di Progetto:** alcuni errori ricorrenti sono dovuti alla non piena conoscenza, da parte dei redattori, delle norme riguardanti la stesura dei documenti (vedi sezione 5.5) in particolare:
 - Inserito il “.” oppure nessun simbolo di punteggiatura al posto del “;” negli elenchi puntati al termine di ogni voce interna;
 - Errato utilizzo del markup nei riferimenti delle estensioni ai file;
 - Inserimento lettera maiuscola su titoli paragrafi ove non necessario;
 - Mancanza, nei termini di glossario, della marcatura “G” in pedice;
 - Non utilizzato il corsivo per il riferimento ad altri documenti.
- **Errori ortografici:**
 - Eccessivo utilizzo del punto e virgola nei periodi. È preferibile inserire il punto;
 - Dovuti a distrazioni e/o errori di battitura;
 - Punteggiatura sbagliata o mancante.
- **Errori riguardanti la lingua italiana:**
 - Passaggio da un tempo verbale ad un altro nello stesso periodo;
 - Utilizzo di parole il cui significato attribuito dal redattore, non è conforme a quello dato dalla lingua italiana;
 - Inserimento della virgola tra soggetto e verbo. Rende poco comprensibile il periodo.