



Norme di Progetto

Informazioni sul documento

Nome documento	Norme di Progetto
Versione	v1.0.0
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
Lista distribuzione	<ul style="list-style-type: none">• <i>7Monkeys</i>• <i>Prof. Tullio Vardanega</i>• <i>Prof. Riccardo Cardin</i>
Uso	Interno

Sommario

Documento contenente l'insieme di norme stabilite dal gruppo *7Monkeys* per la realizzazione di Romeo.

Diario delle Modifiche

Modifica	Autore & Ruolo	Data	Versione
<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
2	Comunicazioni	2
2.1	Composizione email	2
2.1.1	Mittente	2
2.1.2	Destinatario	2
2.1.3	Oggetto	2
2.1.4	Corpo	2
2.1.5	Allegati	3
2.2	Comunicazioni esterne	3
2.3	Comunicazioni interne	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	Struttura	5
4.1.1	Repository della documentazione	5
4.1.2	Repository del codice	5
4.2	Utilizzo del repository	5
5	Documenti	6
5.1	Versionamento	6
5.2	Template	6
5.3	Struttura dei documenti	6
5.3.1	Prima pagina	6
5.3.2	Diario delle modifiche	7
5.3.3	Indici	7
5.3.4	Formattazione di una pagina	7
5.4	Classificazione dei documenti	7
5.4.1	Documenti informali	7
5.4.2	Documenti formali	7
5.4.3	Verbali	8
5.5	Norme tipografiche	8
5.5.1	Stile di testo	8
5.5.2	Punteggiatura	9
5.5.3	Composizione del testo	9
5.5.4	Formati ricorrenti	9
5.6	Componenti grafiche	10
5.6.1	Immagini	10
5.6.2	Tabelle	10
5.7	Procedura di avanzamento di un documento	10
6	Glossario	11
6.1	Inserimento dei termini	11
6.2	Eliminazione dei termini	11
7	Analisi dei requisiti	12
7.1	Requisiti	12
7.2	Casi d'uso	12

8	Progettazione	13
8.1	Diagrammi	13
8.2	Design pattern	13
8.3	Classi di verifica	13
8.4	Stile di progettazione	13
9	Codifica e convenzioni	14
9.1	Linguaggio di codifica	14
9.2	Convenzioni di codifica	14
9.2.1	Nomenclatura	14
9.2.2	Intestazione di un file	14
9.2.3	Commenti nei metodi	15
9.2.4	Commenti nei file di implementazione	15
9.3	Convenzioni	15
10	Protocollo per la gestione del progetto	16
10.1	Creazione milestone	16
10.2	Creazione ticket	16
10.3	Esecuzione dei compiti	17
10.4	Chiusura della milestone	17
11	Ambiente di Lavoro	18
11.1	Sistema operativo	18
11.2	Coordinamento	18
11.2.1	Repository Git	18
11.2.2	Dropbox	18
11.2.3	Google Drive	18
11.2.4	Google Calendar	19
11.3	Ambiente documentale	19
11.3.1	Stesura documenti	19
11.3.2	Verifica ortografica	19
11.3.3	Pianificazione delle attività	19
11.3.4	Documentazione del codice	19
11.4	Ambiente di sviluppo	19
11.4.1	Framework	19
11.4.2	Ambiente di codifica	19
11.5	Diagrammi UML	20
11.6	Verifica	20
11.6.1	Tracciamento dei requisiti	20
11.6.2	Verifica dei documenti	20
11.6.3	Verifica dei diagrammi	20
11.6.4	Verifica del codice	20
11.7	Scripts	21
A	Lista di controllo	22

Elenco delle figure

1	Procedura di avanzamento di un documento.	11
2	Interfaccia di GitHub per la creazione di una milestone.	16
3	Interfaccia di GitHub per la creazione di un ticket.	17

1 Introduzione

1.1 Scopo del documento

Questo documento ha lo scopo di fissare le norme che tutti i membri del gruppo *7Monkeys* 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 v1.0.0* 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.

2 Comunicazioni

2.1 Composizione email

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

2.1.1 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 *nessariamente* 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.

2.1.2 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 committenti del progetto;
- **Interna:** l'unico indirizzo utilizzabile è 7monkeys.swe@gmail.com.

Sono ammesse alcune eccezioni:

- **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 personale¹;
- **Proposta al Responsabile di Progetto:** nel caso in cui un membro del gruppo voglia richiedere una riunione, dovrà contattarlo al suo indirizzo di posta personale²;
- **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 personali³.

2.1.3 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.

2.1.4 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.

¹Sarà possibile trovare i vari recapiti e-mail personali nel documento “contatti” condiviso su Google Drive.

²Vedi nota precedente.

³Vedi due note precedenti.

2.1.5 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.

2.2 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 7Monkeys intrattenendo le corrispondenze con i proponenti e i committenti.

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.

2.3 Comunicazioni interne

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

Quando un membro del gruppo vuole inviare una mail a tutti i componenti, deve inviare il messaggio dalla sua mail personale verso la mail 7monkeys.swe@gmail.com. Un inoltrato automatico provvederà a trasmettere la mail 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 videochiamate.

È necessario redigere un verbale⁴ nel caso in cui siano state prese decisioni o siano emersi dettagli inerenti allo sviluppo del progetto.

⁴Vedi sezione 5.4.3 per maggiori dettagli.

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 2.2 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 *7Monkeys*.

Per il repository_G è stato scelto di utilizzare il servizio GitHub_G⁶, il quale utilizza il sistema di versionamento Git_G.

4.1 Struttura

4.1.1 Repository della documentazione

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

All'interno di questo repository_G andranno messi tutti i file riguardante la documentazione.

Questa cartella di base contiene le seguenti sottocartelle:

- **Documentazione:** contiene il template $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ e l'insieme di documenti realizzati dal gruppo. Questa cartella contiene le seguenti sottocartelle:
 - **Template:** contiene il template $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$;
 - **Nome_del_Documento:** ogni documento in fase di redazione dovrà essere posizionato assieme a tutti i file di cui necessita. Se il documento necessita di immagini, esse dovranno essere salvate in una sotto-cartella dal nome “Immagini”;
 - **Verbali:** contiene l'insieme di verbali redatti dal gruppo.
- **Scripts:** contiene l'insieme di scripts utilizzati per automatizzare alcune procedure;
- **Revisioni:** contiene tutti i PDF_G formali da consegnare alle revisioni. I file saranno raggruppati in cartelle aventi come nome l'acronimo della revisione a cui fanno riferimento. Nello specifico la struttura delle sottocartelle sarà la seguente:
 - **RR:** Revisione dei Requisiti;
 - **RP:** Revisione di Progettazione;
 - **RQ:** Revisione di Qualifica;
 - **RA:** Revisione di Accettazione.

4.1.2 Repository del codice

L'indirizzo root del repository_G del codice è: <https://github.com/nicolobissacco/7MonkeysCode>.

La struttura del repository_G del codice verrà decisa durante la fase di progettazione.

4.2 Utilizzo del repository

Per avere la massima sincronizzazione e avere una coerenza con le parti sviluppate dagli altri componenti del gruppo è *indispensabile* eseguire le operazioni di sincronizzazione all'inizio e alla fine di ogni sessione di lavoro.

In particolare andranno eseguite le seguenti azioni:

- **Pull:** prima di iniziare la sessione di lavoro, per ottenere i file aggiornati;
- **Commit:** al termine di una sessione di lavoro, per commentare le modifiche effettuate;
- **Push:** subito dopo l'operazione di commit, per caricare le modifiche nel repository_G remoto;
- **Merge:** se durante la push sorgono dei problemi per modifiche effettuate da un altro membro.

I commenti aggiunti al commit dopo la modifica di un file devono essere chiari e specificare le modifiche effettuate. Per garantire una buona leggibilità dei commenti, i commit devono essere effettuati per ogni file modificato o gruppo di files nello stesso contesto.

Inoltre per evitare conflitti è *sconsigliato* lavorare sugli stessi file contemporaneamente.

⁶<https://github.com>

5 Documenti

In questa sezione si presentano i vari standard adottati dal gruppo *7Monkeys* nello scrivere i documenti prodotti durante il processo di sviluppo del software.

5.1 Versionamento

In ogni documento deve essere specificata la versione, la quale va definita nel seguente modo:

$$X.Y.Z$$

dove:

- **X:** identifica la versione del rilascio, ogni incremento causa l'azzeramento dei valori di Y e Z. In particolare la versione di un documento alla Revisione dei Requisiti sarà la 1.0.0, alla Revisione di Progettazione sarà la 2.0.0 e via dicendo per le successive revisioni;
- **Y:** l'incremento di tale parametro indica l'avvenuta verifica dello stesso, la prima versione del documento avrà parametro Y uguale a zero. Ogni incremento del parametro Y comporta l'azzeramento del parametro Z;
- **Z:** identifica la revisione del documento, ad ogni aggiunta, modifica o correzione comporta un incremento del parametro Z.

5.2 Template

Ogni documento deve essere realizzato utilizzando il template L^AT_EX presente nel repository⁶. Per maggiori informazioni su come utilizzare il template consultare il seguente link: <https://github.com/nicolobissacco/7MonkeysDoc/blob/master/Documentazione/Instructions/instructions.pdf>. Inoltre è stato creato un elenco di comandi L^AT_EX per facilitare la scrittura di parole ricorrenti⁷.

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 documento⁸;
- 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;
- Tipo d'uso del documento⁹;
- La lista di distribuzione del documento;
- Un sommario, contenente una breve descrizione del documento.

⁶Consultare https://github.com/nicolobissacco/7MonkeysDoc/blob/master/Documentazione/Command_List/commands_list.pdf.

⁸Espressa come indicato in sezione 5.1.

⁹Potrà essere interno o esterno.

5.3.2 Diario delle modifiche

Nella seconda pagina di ogni documento deve essere presente il diario delle modifiche.

Ogni riga del 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 documento¹⁰;
- 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 cartella¹¹.

Tali documenti devono essere rinominati osservando le seguenti regole:

- 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*¹² e sono pronti per essere rilasciati.

Quando un documento diventa formale sarà possibile che venga 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”.

Un esempio di nome è il seguente: *Studio_di_Fattibilita-v1.0.0.pdf*.

¹⁰La data deve essere espressa come riportato in sezione 5.5.4.

¹¹Per maggiori dettagli vedere la sezione 4.1.1.

¹²Fatta eccezione per il *Piano di Progetto*, che viene approvato da un qualsiasi altro membro del gruppo.

5.4.3 Verbali

Per verbali si intendono tutti quei documenti redatti come promemoria in seguito ad un incontro. Tutti i verbali esterni saranno redatti dal *Responsabile di Progetto*, mentre i verbali interni saranno redatti da un qualsiasi membro presente all'incontro.

I verbali dovranno essere nominati secondo il seguente criterio:

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

dove:

- **Numero del verbale:** indica un numero progressivo che identifica il verbale¹³;
- **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'incontro¹⁴.

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

- **Data**¹⁵
- **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 sottosezione per tutte le informazioni riguardanti l'ortografia, la tipografia e l'assunzione di uno stile uniforme in tutti i documenti.

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*);
 - **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.

¹³Tale numero dovrà partire da uno.

¹⁴La data dovrà essere formattata come indicato in sezione 5.5.4.

¹⁵Formato espresso come 5.5.4

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 `\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*¹⁶.

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 8601¹⁷:

$$AAAA - MM - GG$$

dove:

- **AAAA:** rappresenta il formato dell'anno scritto con tutte e quattro le cifre;
- **MM:** rappresenta il mese scritto con esattamente due cifre¹⁸;
- **GG:** rappresenta il giorno scritto con esattamente due cifre.¹⁹
- **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 v1.0.0*);
- **Nomi propri:** i nomi propri di persona devono essere scritti nel formato “Cognome Nome”;
- **Nome del gruppo:** ci si riferirà al gruppo 7Monkeys con la dicitura “7Monkeys”, 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:

¹⁶Le occorrenze dei termini presenti nei titoli, nelle note a piè di pagina e negli oggetti degli elenchi non andranno evidenziati.

¹⁷http://it.wikipedia.org/wiki/ISO_8601

¹⁸Nel caso lo si possa scrivere utilizzando solo una cifra, si impone di anteporre uno zero al numero.

¹⁹Vedi la nota precedente.

- 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 **PDF_G** o **PNG_G**, è consigliato utilizzare **PDF_G** vettoriale in quanto la sua risoluzione non dipende dal ridimensionamento dell'immagine.

La conversione di un immagine in formato **PDF_G** è consentita grazie all'utilizzo del software Gimp²⁰, il cui funzionamento è conosciuto da tutti i componenti del gruppo *7Monkeys*.

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 figura²¹.

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 tabella²².

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

5.7 Procedura di avanzamento di un documento

Ogni qualvolta un redattore ritiene che la stesura di un documento sia terminata andrà seguita la seguente procedura²³:

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*.

²⁰<http://www.gimp.org>

²¹Con breve si vuole intendere che la descrizione deve stare in una riga.

²²Vedi nota precedente.

²³In figura 1 si vede un diagramma che rappresenta la procedura.

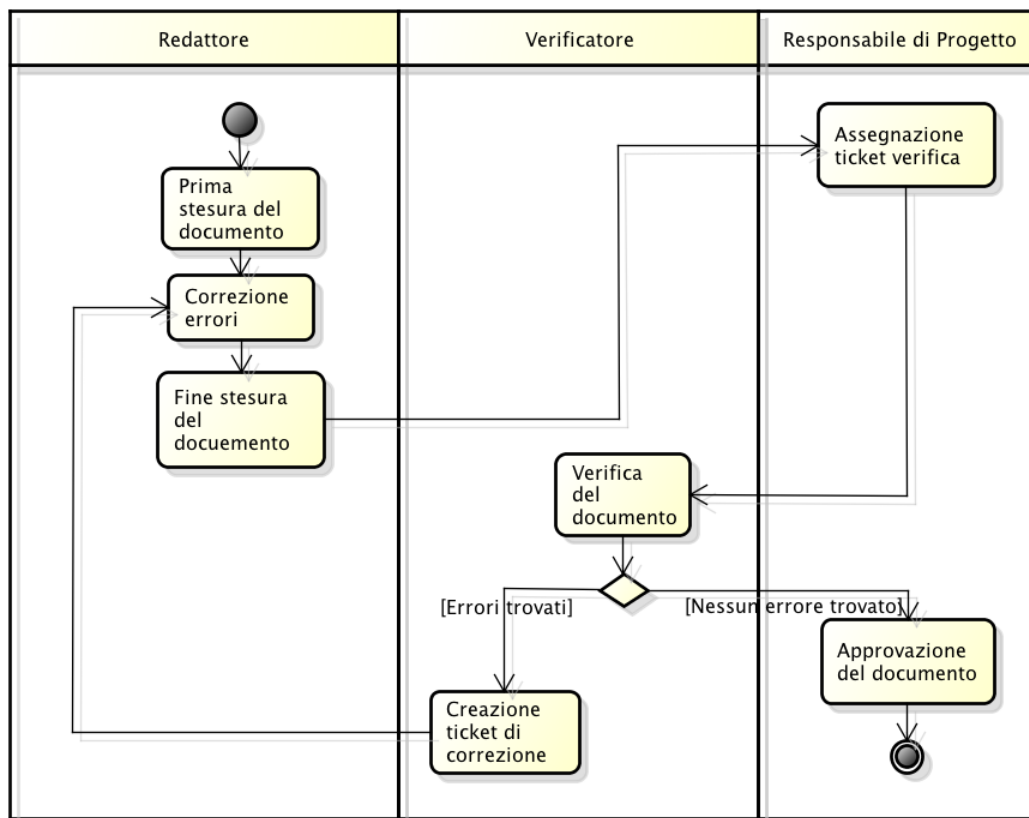


Figura 1: Procedura di avanzamento di un documento.

6 Glossario

Il *Glossario* sarà un documento unico e riassuntivo per tutti i documenti e conterrà, in ordine lessicografico, una descrizione dell'insieme di parole o acronimi che possono generare ambiguità o confusione.

Ogni occorrenza di un termine presente nel *Glossario* è indicato da una “G” in pedice.

6.1 Inserimento dei termini

Data l'universalità del *Glossario* rispetto a tutti i documenti redatti dal gruppo 7Monkeys, ogni membro del gruppo potrà apportarne delle modifiche, tranne il *Verificatore* del documento.

Per l'inserimento di un termine nel *Glossario* si dovrà seguire la seguente procedura:

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

6.2 Eliminazione dei termini

Ogni membro del gruppo per richiedere l'eliminazione di un termine del *Glossario* dovrà seguire la seguente procedura:

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

7 Analisi dei requisiti

Il documento *Analisi dei Requisiti* sarà redatto dagli *Analisti*.

In tale documento dovranno essere presenti tutti i requisiti e i casi d'uso_G emersi dal capitolato o dall'Analisi dei Requisiti.

7.1 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:

$$R\{\text{importanza}\}\{\text{tipo}\}\{\text{codice}\}$$

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.

Ogni requisito andrà inserito, nel documento di *Analisi dei Requisiti*, in una tabella contenente l'identificativo, una breve descrizione del requisito e la fonte.

7.2 Casi d'uso

Per ogni caso d'uso_G andrà fornito:

- **Identificativo:** Ogni caso d'uso_G va identificato da un codice composta dalle due lettere maiuscole "UC", seguite dal numero identificativo del caso d'uso_G;
- **Diagramma:** andrà utilizzato UML_G per creare dei diagrammi di caso d'uso_G che semplificano la comprensione dei requisiti.

Di seguito sono riportate delle convenzioni adottate nella realizzazione dei diagrammi:

- **Standard:** lo standard UML_G utilizzato è il 2.0;
- **Lingua:** la utilizzata all'interno dei diagrammi è l'italiano.
- **Descrizione:** una breve didascalia che descrive il caso d'uso_G.
- **Attori:** gli attori coinvolti nel caso d'uso_G;
- **Precondizione:** la precondizione del requisito;
- **Postcondizione:** la postcondizione del requisito.

8 Progettazione

Terminata la fase di Analisi inizierà la fase di Progettazione. Durante la fase di Progettazione i *Progettisti* dovranno aderire alle seguenti specifiche.

8.1 Diagrammi

Andrà utilizzato il linguaggio UML_G per definire i seguenti diagrammi:

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

8.2 Design pattern

I *Progettisti* dovranno utilizzare i design pattern_G più adatti nel contesto, in modo da rendere l'applicazione il più efficiente possibile.

8.3 Classi di verifica

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

8.4 Stile di progettazione

Durante la fase di Progettazione bisognerà fare attenzione a:

- **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.

9 Codifica e convenzioni

9.1 Linguaggio di codifica

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

9.2 Convenzioni di codifica

Di seguito è riportato l'insieme di norme e convenzioni che il gruppo *7Monkeys* seguirà nella scrittura e documentazione del codice.

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

9.2.1 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).

9.2.2 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.

9.2.3 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;

9.2.4 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.

9.3 Convenzioni

Per rendere il codice più leggibile e facilmente comprensibile si è deciso che 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*.

10 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.

10.1 Creazione milestone

Il *Responsabile di Progetto* dovrà provvedere alla creazione di una milestone_G per la prossima revisione a cui il gruppo *7Monkeys* 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 richiesti²⁴.

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 new milestone' form. On the left, there is a 'Title' input field and a larger 'Description' text area. On the right, there is a 'Due Date (optional)' section with a 'clear' link and a calendar widget. The calendar is for December 2013, with the 7th of the month highlighted in blue.

Figura 2: Interfaccia di GitHub per la creazione di una milestone.

10.2 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.

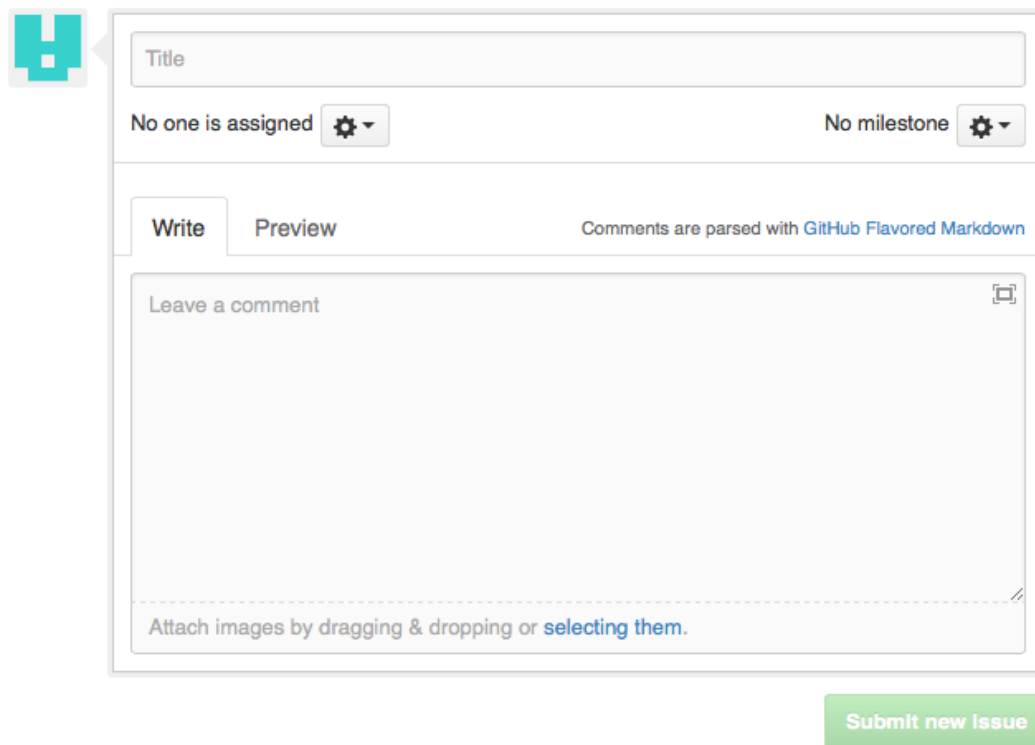
Ogni ticket dovrà essere assegnato ad un unico membro del team *7Monkeys* e dovrà avere le seguenti caratteristiche²⁵:

- **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*;

²⁴I possibili cambi da completare sono riportati in figura 2.

²⁵In figura 3 è mostrata l'interfaccia per la creazione di ticket.

- 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.



The image shows the GitHub 'New Issue' form. At the top left is the GitHub logo. The form has a 'Title' input field. Below it, there are two status indicators: 'No one is assigned' with a gear icon and 'No milestone' with a gear icon. Below these are two tabs: 'Write' (active) and 'Preview'. To the right of the tabs, it says 'Comments are parsed with GitHub Flavored Markdown'. The main area is a large text input field with the placeholder 'Leave a comment'. At the bottom of this field, there is a dashed line and the text 'Attach images by dragging & dropping or selecting them.' To the right of the text input field is a small icon of a document with a plus sign. At the bottom right of the form is a green button labeled 'Submit new issue'.

Figura 3: Interfaccia di GitHub per la creazione di un ticket.

10.3 Esecuzione dei compiti

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.

10.4 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.

11 Ambiente di Lavoro

In questa sezione verrà illustrato in dettaglio l'ambiente di lavoro che il team *7Monkeys* utilizzerà durante lo sviluppo del progetto Romeo.

11.1 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.

11.2 Coordinamento

Il coordinamento del gruppo avviene tramite:

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

11.2.1 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.

Per maggiori dettagli sul repository_G e sulle sue norme di utilizzo vedi sezione 4.

11.2.2 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.

11.2.3 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.

11.2.4 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.

11.3 Ambiente documentale

11.3.1 Stesura documenti

Per la scrittura dei documenti è stato scelto di utilizzare il linguaggio di markup_G L^AT_EX²⁶. Come editor è consigliato l'utilizzo di T_EXmaker²⁷, il quale è disponibile per tutti i principali sistemi operativi.

11.3.2 Verifica ortografica

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 dizionario.

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

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

11.3.3 Pianificazione delle attività

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

11.3.4 Documentazione del codice

Per la documentazione del codice è stato scelto di usare il sistema Doxygen²⁸, 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 L^AT_EX.

11.4 Ambiente di sviluppo

11.4.1 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, che permette il supporto a varie lingue;
- **Qt Designer:** è presente questo tool, il quale facilita la creazione di interfacce;
- **Esperienza:** tutti i membri del team hanno già utilizzato il framework_G Qt_G in precedenza.

La versione di riferimento delle librerie Qt_G è la 5.1.

11.4.2 Ambiente di codifica

Per la scrittura del codice è stato scelto di utilizzare l'IDE_G Qt_G Creator.

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

Inoltre Qt_G Creator si integra con Git_G.

La versione di Qt_G Creator utilizzata è la 2.8.1.

²⁶<http://latex-project.org/ftp.html>

²⁷<http://www.xmlmath.net/texmaker/>

²⁸<http://www.stack.nl/~dimitri/doxygen/>

11.5 Diagrammi UML

Per la realizzazione dei diagrammi UML_G è stato scelto Astah_G Professional Edition. Per richiedere gratuitamente la versione professional si visita il sito <http://astah.net/student-license-request> si compilano i campi richiesti e si seguono le istruzioni, la licenza arriverà via email.

11.6 Verifica

11.6.1 Tracciamento dei requisiti

Per il tracciamento dei requisiti l'*Amministratore di Progetto* ha creato un'applicazione web che ad ogni caso d'uso_G ne associa i vari requisiti e viceversa.

11.6.2 Verifica dei documenti

Il processo di verifica dei documenti viene eseguito ogni qualvolta un documento abbia subito una modifica e deva 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 Aspell_G 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 v1.0.0*, 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 10.2.

11.6.3 Verifica dei diagrammi

Al *Verificatore* è richiesta la verifica dei seguenti diagrammi UML_G:

- **Diagrammi di flusso:** la verifica dei diagrammi di flusso deve avvenire manualmente controllando che essi aderiscano allo standard UML_G 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 UML_G 2.0, ci sia un corretto utilizzo di inclusioni ed estensioni e ci sia la massima corrispondenza tra il diagramma e la descrizione testuale.

11.6.4 Verifica del codice

- **Analisi statica:** per l'analisi del codice si useranno i seguenti strumenti: Per l'analisi statica del codice si useranno i seguenti strumenti:
 - **Compiler:** il compilatore è il primo strumento di analisi, in quanto segnala in modo molto preciso errori e warning. In particolare il progetto, dovendo essere multiplatforma, dovrà compilare con i seguenti compilatori:

- * **GCC**²⁹: compilatore per ambiente Linux_G;
- * **Clang**³⁰: compilatore per ambiente Mac OS_G;
- * **MinGW**³¹: compilatore per ambiente Windows_G.
- **Test**: Per i test di unità sul codice si utilizzerà il framework_G CppUnit³². Per maggiori dettagli si rimanda al documento *Piano di Qualifica v1.0.0*;
- **Analisi dinamica**: L'analisi dinamica verrà effettuata con:
 - **Valgrind**: tool 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**: tool per il controllo della memoria in ambiente Windows_G;
 - **QtTest**: libreria integrata nella suite Qt_G, strumento molto comodo per eseguire gli Unit Test³³.

11.7 Scripts

Per automatizzare 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 RR` verranno generati tutti i PDF_G e salvati sulla cartella RR;
- **Fare il controllo ortografico**: tramite il comando `make aspell` verrà invocato l'applicativo aspell_G su tutti i documenti;
- **Eliminare i file generati**: tramite il comando `make clean` verranno eliminati tutti i file generati da vecchie compilazioni.

²⁹<http://gcc.gnu.org>

³⁰<http://clang.llvm.org>

³¹<http://www.mingw.org>

³²http://sourceforge.net/apps/mediawiki/cppunit/index.php?title=Main_Page

³³<http://qt-project.org/doc/qt-5.0/qttestlib/qtest-overview.html>

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;
 - Mancanza, nei termini di glossario, della marcatura “G” in pedice;
 - Non utilizzato il corsivo per il riferimento ad altri documenti.
- **Errori ortografici:**
 - 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.