

NORME DI PROGETTO

Informazioni

|  |  |  |  |
| --- | --- | --- | --- |
| Modifica | Autore | Data | Versione |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
| Sezione documentazione | Giacomo Pinato | 26/11/2013 | 0.0.4 |
| Sezione comunicazioni | Giacomo Pinato | 26/11/2013 | 0.0.3 |
| Sparo cazzare |  |  |  |
| Stesura Iniziale | Giacomo Pinato | 26/11/2013 | 0.0.1 |

# 

[1 Introduzione](#h.hgpz2qevjczq)

[1.1 Scopo del documento](#h.eiabwe4r7hch)

1.2 Scopo del prodotto

1.3 Riferimenti

1.3.1 Normativi

1.3.2 Bibliografici

2 [Ruoli di progetto](#h.jjtkzpvq32xy)

2[.1 Responsabile di Progetto](#h.9axzg1816c3v)

2[.2 Amministratore](#h.g25hnfibjyuh)

2[.3 Analista](#h.bs2zn3pwuwcq)

2[.4 Progettista](#h.dqv24ojnv0q5)

2[.5 Verificatore](#h.i5i1e897dopp)

2[.6 Programmatore](#h.rjuciyhgzry0)

3 [Comunicazioni](#h.gmlry2juipw0)

3[.1 Email](#h.9utnteajya7b)

3[.2 Comunicazioni Interne](#h.lfa7df3ollmu)

3[.2.1 Skype](#h.1h07w27la4a5)

3[.2.2 Mailing List](#h.89mewtv2lj5i)

3[.3 Riunioni](#h.8vdl8ht8yb9l)

3[.3.1 Richiesta](#h.kiccvanshdx9)

3[.3.2 Svolgimento](#h.tforurbpee0p)

3[.3.3 Verbale](#h.8gdlb23hhdqi)

3[.3.3.1 Struttura del Verbale](#h.57m7ufqdecoy)

3[.4 Comunicazioni con i committenti e proponenti](#h.vigx17ab8064)

3[.4.1 Prerequisiti](#h.ue37qdm2wm8n)

3[.4.2 Richiesta di colloquio](#h.dct27tqpqfva)

3[.4.3 Verbale](#h.i0sjdodt1i43)

4 [Ambiente e strumenti di lavoro](#h.nb68o6i429nh)

4[.1 Redmine](#h.3v275g5h6g21)

4[.2 Google Calendar](#h.7uykragxw9i1)

4[.2 Repository](#h.27s8ltkobowt)

4[.2.1 Branch](#h.dyu7olxx6kip)

4[.2.1.1 Master](#h.t1h84nf7couk)

4[.2.1.2 Branch Secondari](#h.kcbikzxondah)

4[.3 Integrazione Continua](#h.y37zrcc4rlh1)

5 [Documentazione](#h.3n2jd0x44f9o)

5[.1 Template](#h.3f8cxq2px667)

5[.2 Struttura del documento](#h.p0v22r7xp8ah)

5[.2.1 Header](#h.t4vzsj33fhw0)

5[.2.2 Footer](#h.oiis98xsrzrv)

5[.2.3 Prima pagina](#h.nl8f3tc2jml)

5[.2.4 Seconda pagina](#h.hcbqzupkdbno)

5[.2.5 Terza pagina](#h.plv27o38u7h2)

5[.3 Documentazione del codice](#h.mada0gleajnr)

5[.3.1 Header dei file](#h.2smla2sflj9k)

5[.3.2 Documentazione dei metodi](#h.1r34yix4ydh9)

5[.3.3 Documentazione delle classi](#h.fwht8i1zrgp7)

5[.4 Norme tipografiche e stili di testo](#h.p9w7a8o3gckg)

5[.5 Sigle e abbreviazioni](#h.y4zm6720rjmb)

5[.6 Termini del Glossario](#h.crgkctit9z4n)

5[.7 LaTeX](#h.l8e8ujuhcxxk)

6 [Versionamento](#h.z42o9lxnwkc7)

6[.1 Regole generali](#h.iomou8wv7h8m)

6[.2 Versionamento del software](#h.qxddka3nvcnf)

6[.2 Versionamento dei documenti](#h.wzhb8nspoiei)

6[.3 Tabella delle modifiche](#h.12mehj78716s)

7 [Analisi dei requisiti](#h.grybm76v79ul)

7[.1 Identificazione e classificazione dei requisiti](#h.xf06icytxle4)

7[.2 Casi d’uso e UML](#h.z6ukl0tyt2jz)

8 [Norme di sviluppo](#h.388osbobgey3)

[8.1 Creare un nuovo progetto](#h.kmiso0794tis)

[8.2 Ticket](#h.3yvdkj22akyb)

[8.2.1 Creazione dei ticket](#h.j14uae6qjv61)

[8.2.2 Tipologie di Ticket](#h.jgvl7j9mdbe4)

[8.2.2.1 Ticket di pianificazione](#h.c9v9cb4xqpst)

[8.2.2.2 Ticket di realizzazione e controllo](#h.iu6xjhvtchda)

[8.2.2.3 Ticket di verifica](#h.uioxyub5cw9x)

[8.2.2.4 Ticket di modifica](#h.ehok1e14lalj)

[9 Glossario](#h.9h53e2dg3zjx)

[11.1 Analisi statica](#h.xaghgtmrzttv)

[11.1.1 Walkthrough](#h.rzuwjzmr2x7a)

# 

# 

# 1 Introduzione

## 1.1 Scopo del documento

Questo documento è volto a definire le norme che dovranno essere osservate da tutti i componenti del team per l’intera durata del progetto. Tali norme sono volte a garantire la qualita’ finale del prodotto attraverso la rigida regolamentazione dei processi e delle strategie di produzione. In queste pagine vengono delineate le linee guida e le norme per tutte le attivita’ che concorreranno allo sviluppo del software, della documentazione e del progetto in generale.

## 1.2 Scopo del prodotto

Lo scopo del progetto è produrre un framework per generare interfacce web di amministrazione dei dati di business basati sullo stack Node.js e MongoDB.

L’obiettivo e’ quello di semplificare il lavoro allo sviluppatore che dovra’ rispondere in modo rapido e standard alla richieste degli esperti di business.

# 2 Ruoli di progetto

Per la piena riuscita del progetto è indispensabile distinguere i vari ruoli che concorrono alla creazione del prodotto finale e le loro diverse responsabilità e competenze.

Ogni ruolo avrà una specifica area di competenza, degli specifici compiti e oneri e delle particolari autorizzazioni. Ogni componente dovrà limitarsi ai compiti ad esso assegnati e, nel caso qualcosa esuli dal suo campo di pertinenza, lo stesso dovrà rivolgersi al suo collega attualmente occupante il ruolo competente.

Tutti i membri, a rotazione, dovranno occupare come minimo una volta ciascuno dei ruoli descritti sottostante.

## 2.1 Responsabile di Progetto

Il Responsabile di Progetto incentra su di sé le responsabilità di scelta ed approvazione dei lavori. Ha inoltre il ruolo di rappresentare il gruppo nei contatti con l’esterno e durante la presentazione dei lavori.

Le sue competenze principali comprendono

• Pianificazione, coordinamento e controllo delle attività;

• Gestione e controllo delle risorse;

• Approvazione delle analisi di gestione e rischio;

• Approvazione dei documenti;

• Comunicazioni con i committenti/proponenti.

Il responsabile ha il compito di assicurarsi che le attività di verifica vengano svolte sistematicamente seguendo le Norme di Progetto, che vengano rispettati i ruoli e le competenze assegnate nel Piano di Progetto e che non vi siano conflitti di interesse tra redattori e verificatori. Ha inoltre l’onere di gestire la creazione e l’assegnazione dei ticket di pianificazione e di assegnare ad un membro del gruppo il ruolo di responsabile di quest’ultimo, nel caso riguardi una sotto-attività.

## 2.2 Amministratore

L’Amministratore è il responsabile del controllo, dell’efficienza e dell’operatività dell’ambiente di lavoro e degli strumenti per la condivisone e la sincronizzazione.

* L’individuazione e la gestione di strumenti per automatizzare quanto più possibile processi o attività
* L’individuazione e la gestione di strumenti per il controllo dei processi e delle risorse
* L’individuazione e la gestione di strumenti e strategie per il controllo della qualità
* Gestione del versionamento

## 2.3 Analista

L’analista è il responsabile dell’analisi dei requisiti di progetto. Dopo aver compreso pienamente la natura del problema e tutti i suoi domini, il suo ruolo è delineare vincoli e caratteristiche del prodotto finale, redigendo una specifica di progetto dettagliata, precisa e non ambigua, comprensibile sia dal proponente che dal progettista.

## 2.4 Progettista

Il progettista è colui che disegna una soluzione attuabile ed efficace che soddisfi i requisiti dettati dagli analisti. Il suo compito è progettare un’architettura che assicuri una facile manutenibilità del prodotto e una buona scomposizione in moduli indipendenti tra di loro.

## 2.5 Verificatore

Il Verificatore è responsabile delle attività di verifica. Ha il compito di assicurare che i documenti e il codice rispettino gli standard qualitativi precedentemente definiti utilizzando gli strumenti e i metodi proposti dal Piano di Qualifica e attenendosi a quanto descritto nelle Norme di Progetto.

## 2.6 Programmatore

Il Programmatore è responsabile delle attività di codifica e delle componenti di usilio

necessarie per l’esecuzione delle prove di verifica e validazione. Le responsabilità di tale ruolo sono:

• Implementare rigorosamente le soluzioni descritte dal Progettista per la realizzazione del progetto

• Scrivere codice e la sua relativa documentazione che rispettino gli standard stabiliti per la loro scrittura

• Implementare i test sul codice scritto, necessari per prove di verifica e validazione.

# 3 Comunicazioni

## 3.1 Email

La email ufficiale del gruppo è [ApertureSWE@gmail.com](mailto:ApertureSWE@gmail.com).

Tale mail può essere usata soltanto dal Responsabile di progetto ed è da utilizzarsi per tutte le comunicazioni che il gruppo terrà con l’ambiente esterno.

## 3.2 Comunicazioni Interne

### 3.2.1 Skype

Per facilitare le comunicazioni instantanee è stata creata su skype una chat di gruppo dove sono presenti tutti i membri del team di sviluppo. Tale chat deve essere usata solo per conversazioni non ufficiali o per discussioni non importanti come lo scambio di articoli, consigli o comunicazioni non rilevanti.

### 3.2.2 Mailing List

E’ stata creata una mailing list all’indirizzo [aperture-team@googlegroups.com](mailto:aperture-team@googlegroups.com).

Ogni email inviata a tale indirizzo verrà inoltrata alla mail personale di ogni componente nel gruppo.

La mailing list è collegata ad un gruppo su Google Groups, che permette di velocizzare le comunicazioni implementando una simil-chat tramite email distribuite. Mantiene inoltre uno storico di qualsiasi comunicazione. Va usato quindi come strumento per discussioni e comunicazioni ufficiali.

## 3.3 Riunioni

### 3.3.1 Richiesta

Ogni membro del team può richiedere che venga organizzata una riunione al Responsabile, che verificata la motivazione della richiesta, accoglie o meno la stessa.

Il Responsabile di progetto ha la facoltà di indire riunioni qualora sentisse la necessità di farlo.

La riunione deve essere segnalata sul calendario di gruppo con almeno due giorni di anticipo e non deve andare a sovrapporsi con impegni precedenti di altri componenti, a meno che la loro presenza non possa essere esclusa. Deve inoltre essere inviata una mail circolare di promemoria a tutti i membri del gruppo contente giorno, ora e motivazione della riunione.

### 3.3.2 Svolgimento

Alle riunioni è gradita, ma non richiesta, la partecipazione di tutti i membri del gruppo. E’ giustificata l’assenza nel caso in cui la riunione riguardi temi non inerenti al ruolo di progetto che si sta ricoprendo o nel caso di impegni validi, giustificabili e improrogabili.

### 3.3.3 Verbale

Ad ogni riunione verrà eletto un segretario che avrà il compito di annotare gli argomenti di discussione e le decisioni prese durante la stessa.

Dovrà inoltre redigere un verbale che verrà ufficialmente raccolto e archiviato come documentazione interna entro tre giorni dalla data della riunione.

### 3.3.3.1 Struttura del Verbale

Ogni verbale dovrà essere strutturato come segue:

* Data
* Luogo
* Ora
* Durata
* Partecipanti interni
* Partecipanti esterni
* Motivazione della riunione/incontro
* Argomenti trattati
* Varie ed eventuali

## 3.4 Comunicazioni con i committenti e proponenti

### 3.4.1 Prerequisiti

Prima di richiedere un colloquio personale con i proponenti e/o committenti il team deve preparare un documento che riassume i punti che verranno discussi contenente argomenti, dubbi e domande da porre.

### 3.4.2 Richiesta di colloquio

I colloqui con i committenti di progetto possono essere richieste dal Responsabile mediante la mail ufficiale del team. Tutti i membri devono essere avvisati prima di richiedere un incontro con il committente.

### 3.4.3 Verbale

Alla fine del colloquio deve essere redatto un verbale (vedi 2.3.2) che riassume gli argomenti trattati, le conclusioni, nonchè le strategie a cui si è arrivati in concordanza col committente/proponente.

# 4 Ambiente e strumenti di lavoro

## 4.1 Redmine

Come piattaforma per la gestione del progetto è stato scelto Redmine. Le principali funzioni che esso fornisce sono:

• Un sistema di gestione dei ticket;

• Il grafico Gantt delle attività;

• Un calendario per organizzare i compiti e le attività;

• La visualizzazione del repository associato al progetto;

• Un sistema di gestione del tempo e delle milestone.

### 4.2 Google Calendar

Viene messo a disponsizione del team un calendario per coordinare temporalmente le attività e gli impegni di ciascuno dei componenti.

Tutti i membri del team useranno il calendario messo a disposizione su Google Calendar per segnalare i giorni in cui per loro non sarà possibile lavorare al progetto o partecipare a riunioni e sessioni di lavoro di gruppo. I componenti si impegnano a segnare sul calendario quanto prima qualsiasi impegno o impedimento dovesse sorgere e a controllare periodicamente il suddetto per essere aggiornati su impegni altrui e/o di gruppo.

Il calendario verrà inoltre usato dal Responsabile per la segnalazione di riunioni o sessioni di lavoro di gruppo.

## 4.2 Repository

Viene messo a disposizione un repository Git su GitHub per la gestione e il versionamento di codice e documenti tra i vari membri del gruppo.

Il repository pubblico è disponibile all’indirizzo

https://github.com/ApertureSoftware/AperturePublic.git

### 4.2.1 Branch

Nel repository saranno disponibili vari branch per favorire lo sviluppo del codice da parte degli sviluppatori.

### 4.2.1.1 Master

Il branch principale sara’ chiamato *master* e conterra’ l’ultima versione del software stabile rilasciata. Perche’ una nuova versione possa essere caricata nel branch master, quest’ultima deve compilare senza errori o warning e deve aver superato tutti i test disegnati per verificarne la qualita’.

### 4.2.1.2 Branch Secondari

Gli sviluppatori possono richiedere la creazione di un branch secondario al fine di sviluppare provando nuove strategie di sviluppo senza alterare il branch master. Il responsabile di progetto è responsabile dell’approvazione della richiesta.

## 4.3 Integrazione Continua

Si è scelto di adottare Jenkins per applicare l’integrazione continua allo sviluppo del

progetto. Tale software permette di pianificare ed impostare la compilazione del codice, visualizzare lo stato del codice prodotto ed interagire con il software di versionamento.

# 5 Documentazione

## 5.1 Template

Viene fornito un template in LaTeX per la realizzazione della documentazione, sia interna che esterna, che i membri del gruppo dovranno seguire nella stesura dei documenti.

## 5.2 Struttura del documento

### 5.2.1 Header

Ogni documento ha un header presente in ogni pagina che riporta logo e nome del gruppo sulla sinistra.

### 5.2.2 Footer

Ogni documento ha un footer presente in ogni pagina che riporta il nome e la versione del documento a sinistra e il numero della pagina a destra.

### 5.2.3 Prima pagina

La prima pagina di ogni documento deve riportare nel seguente ordine

* Il logo esteso del gruppo
* Il nome del progetto
* Il nome del documento
* Informazioni sul documento (versione, data creazione, data ultima modifica, stato del documento, uso del documento, i redattori del documento e il loro ruolo, la distribuzione)
* Breve sommario del documento

### 5.2.4 Seconda pagina

La seconda pagina deve riportare il diario delle modifiche apportate al documento dalla sua creazione fino alla versione corrente.

### 5.2.5 Terza pagina

La terza pagina deve riportare l’indice del documento.

## 5.3 Documentazione del codice

### 5.3.1 Header dei file

Ogni file dovrà contenere un header strutturato come segue:

/\*

\* File: Nome del file

\* Module: modulo di appartenenza

\* Author: Autore (indirizzo email dell’autore)

\* Created: Data di creazione

\* Version: Versione corrente

\* Description: Descrizione dettagliata del file

\*

\*

\* Modification History: Tabella dei cambiamenti effettuati sul file

\*/

### 5.3.2 Documentazione dei metodi

Ogni metodo/funzione di codice dovrà essere preceduta da un commento contenente le seguenti informazioni:

/\*

\* Name: Nome del metodo

\* Param: lista del tipo dei parametri

\* Descr: Breve descrizione del comportamento del metodo

\* Return: Cosa ritorna la funzione

\*/

### 5.3.3 Documentazione delle classi

Ogni classe deve essere preceduta da un commento contenente le seguenti informazioni:

/\*

\* Name: Nome della classe

\* Descr: Breve descrizione della classe

\*/

## 5.4 Norme tipografiche e stili di testo

Nella stesura della documentazione si dovranno seguire le seguenti indicazioni:

* I documenti dovranno essere grammaticalmente e sintatticamente corretti e scritti in uno stile fluido e comprensibile;
* Elenco puntato termina con il punto e virgola oppure con il punto se è l’ultimo elemento;
* Un carattere di punteggiatura non deve mai seguire un carattere di spaziatura;
* Il testo racchiuso tra parentesi non deve aprirsi o chiudersi con un carattere di spaziatura e non deve terminare con un carattere di punteggiatura;
* Le lettere maiuscole vanno poste solo dopo il punto, il punto di domanda, il punto esclamativo e all’inizio di ogni elemento di un elenco puntato, oltre che dove previsto dalla lingua italiana. È inoltre utilizzata l’iniziale maiuscola nel nome del team, del progetto, dei documenti, dei ruoli di progetto, delle fasi di lavoro e nelle parole Proponente e Committente.
* Nel caso in cui ci si riferisca ad un documento, il titolo di quest’ultimo dovrà essere scritto in *corsivo* e si dovrà riportare la versione riferita
* I ruoli di progetto (es. *analista*) dovranno essere scritti in corsivo per differenziarli dal riferimento generale alla figura professionale.
* Gli acronimi dovranno essere scritti in lettere maiuscole
* E’ preferibile usare la forma attiva a quella passiva
* Quando possibile usare elenchi puntati invece che frasi
* Usare termini specifici e segnare i termini del glossario con la G in pedice
* Dividere il documento in sezioni e sottosezioni titolate
* Le date dovranno essere espresse nella forma AAAA-MM-GG secondo lo standard ISO 8601:2004
* Le pagine di intestazione dei documenti dovranno essere numerate con numeri romani. La numerazione decimale dovrà essere utilizzata dall’inizio del primo capitolo in poi.

## 5.5 Sigle e abbreviazioni

Le sigle e le abbreviazioni dovranno essere utlizzate solo in contesti in cui lo spazio è limitato come tabelle e diagrammi. Sono previste le seguenti abbreviazioni:

* AdR = Analisi dei Requisiti
* GL=Glossario
* NdP = Norme di Progetto
* PdP = Piano di Progetto
* PdQ = Piano di Qualifica
* SdF = Studio di Fattibilità
* ST = Specifica Tecnica
* RA = Revisione di Accettazione
* RP = Revisione di Progettazione
* RQ = Revisione di Qualifica
* RR = Revisione dei Requisiti
* AS = Aperture Software

## 5.6 Termini del Glossario

Alla prima occorrenza di un termine definito nel glossario, esso dovrà essere marcato con la lettera G in pedice della parola. Verrà marcata solo la prima occorrenza al fine di mantenere il testo più pulito e leggibile.

## 5.7 LaTeX

Per la stesura dei documenti verrà utilizzato il linguaggio di markup LaTeX. Esso rende possibile definite template di documenti standardizzati da poter applicare a qualsiasi contenuto, separando cosi formattazione da contenuto e facilitando il lavoro di stesura. LaTeX inoltre dispone di qualsiasi strumento di formattazione di cui si possa avere bisogno, eliminando il bisogno di strumenti ausiliari alla stesura dei documenti.

# 6 Versionamento

Il versionamento verrà effettuato sia sul codice che sui documenti prodotti dal gruppo al fine di differenziare e rendere immediatamente riconoscibile la fase di sviluppo in cui ci si trova attualmente e mantenere uno storico organizzato dei cambiamenti effettuati.

## 6.1 Regole generali

* Il numero di versionamento deve essere nella forma X.Y.Z, con X,Y e Z numeri interi non negativi. Tutti gli elementi devono salire numericamente di una unità alla volta.
* Ogni qualvolta che una versione viene rilasciata non può più essere effettuato alcun cambiamento ad essa. Ogni modifica sarà inserita nella versione successiva.

## 6.2 Versionamento del software

Nel versionare il software, le tre cifre di versionamento verranno modificate in base ai seguenti parametri

* La prima cifra decimale verrà aumentata nel caso in cui vengano introdotti cambiamenti non retrocompatibili al framework. Possono essere inclusi cambiamenti miniori. Nel caso in cui la prima cifra venga aumentata la seconda e la terza ripartono da 0.
* La seconda cifra decimale verrà aumentata nel caso in cui vengano rilasciate nuove funzionalità retrocompatibili. Deve necessariamente essere aumentata se una qualsiasi funzuinalità pubblica del framework viene marcata deprecata. E’ possibile aumentare la seconda cifra anche in caso di rilascio di nuove funzionalità o miglioramenti sostanziali. Al cambiamento della seconda cifra la terza deve ripartire da 0.
* La terza cifra decimale verrà aumentata nel caso di correzione di errori o altri piccoli cambiamenti retrocompatibili.
* La cifra X a 0 è riservata per lo sviluppo iniziale. Le versioni con X a 0 non sono da considerarsi stabili.
* La versione 1.0.0 definisce la prima versione stabile del framework.

## 6.2 Versionamento dei documenti

Nel versionare il software, le tre cifre di versionamento verranno modificate in base alle seguenti regole

X: indica il numero di uscite formali del documento, diviso come segue:

1. Fase di analisi, si estende fino alla Revisione dei Requisiti
2. Fase di analisi in dettaglio, si estende fino all’ingresso nella fase di progettazione
3. Fase di progettazione architetturale, si estende fino alla Revisione di Progettazione
4. Fase di progettazione di dettaglio e codifica,si estende fino alla Revisione di Qualifica;
5. Fase di verifica e validazione, si estende fino alla Revisione di Accettazione e alla fine del progetto.

Y: indica la fase di sviluppo del documento e varia come segue:

1. Fase di stesura del documento, dove il documento è ancora modificabile
2. Fase di verifica del documento, dove il documento non è più modificabile e sta venendo controllato dal verificatore
3. Documento ultimato e approvato

Sarà compito del responsabile impostare la vesione a X.1.0 nel momento in cui assegnerà il ticket di controllo ai verificatori e impostare la versione a X.2.0 nel caso in cui il documento risulti approvato dagli stessi.

Z: indica il numero di modifiche minori apportare al documento. Aumenta al termine di ogni sessione di lavoro sul documento. Non ha un limite massimo.

## 6.3 Tabella delle modifiche

Ogni documento soggetto a versionamento dovrà riportare in qualche sua parte un diario delle modifiche strutturato nella seguente maniera

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Riassunto della modifica | Autore della modifica & ruolo | Data | Versione | Verificatore |

Le modifiche dovranno essere ordinate cronologicamente dalla più alla meno recente.

# 7. Analisi dei requisiti

La stesura del documento denominato *Analisi dei Requisiti* è compito degli *Analisti*.

In questo documento verranno trovati, analizzati e catalogati tutti i requisiti, impliciti e non, imposti dal progetto.

## 7.1 Identificazione e classificazione dei requisiti

È compito degli Analisti stilare una lista dei requisiti emersi dal capitolato e da eventuali riunioni con il proponente. E’ inoltre loro compito identificare tutti quei requisiti impliciti che per loro natura non vengono specifiati ma devono nondimeno essere individuati e studiati. Tutti questi requisiti dovranno essere classificati per tipo e importanza, utilizzando la seguente codifica:

R[importanza][tipo][codice]

• Importanza può assumere i seguenti valori:

O: Requisito obbligatorio

D: Requisito desiderabile

F: Requisito facoltativo

• Tipo può assumere i seguenti valori:

F: Funzionale

Q: Di qualità

P: Prestazionale

V: Vincolo.

• Codice è il codice che identifica univocamente ogni requisito. La sua regolamentazione è descritta nell’*Analisi dei Requisiti.*

## 7.2 Casi d’uso e UML

Successivamente all’individuazione ed al tracciamento dei requisiti si procede all’analisi dei casi d’uso, denominati nelle sezioni seguenti anche come use case o con l’acronimo UC.

Per realizzare i diagrammi UML dei casi d’uso è stato scelto Astah. Il software era stato consigliato dal Professor Cardin e soddisfa tutte le necessità del team.

Tutti i diagrammi devono rispettare la specifica UML 2.0

# 8. Norme di sviluppo

## 8.1 Creare un nuovo progetto

La creazione del progetto è onere del Responsabile di Progetto.

Il progetto è una macro-attività che verrà suddivisa in molte sotto-attività al fine di semplificare la progettazione, la realizzazione e la verifica delle stesse. Ogni attività sarà a carico di un Responsabile.

## 8.2 Ticket

I ticket sono uno strumento per il tracciamento di attività, funzionalità e problematiche utile per sincronizzare e mantere traccia del lavoro del team.

## 8.2.1 Creazione dei ticket

I ticket vengono creati da

* Responsabile di progetto: Crea i ticket di maggiore importanza per l’avanzamento tra le fasi macroscopiche di progetto.
* Responsabile di sotto-progetto: Crea i ticket minori per lo sviluppo delle caratteristiche e delle funzionalità delle singole componenti del progetto.
* Verificatore: Crea ticket per la segnalazione di problematiche o bug che necessitano di controllo e correzione.

## 8.2.2 Tipologie di Ticket

I ticket rappresentano operazioni e attività che devono essere portate a termine.

### 8.2.2.1 Ticket di pianificazione

Gestiscono la pianificazione delle macro-attività di maggiore importanza. Sono organizzati in una gerarchia con vari livelli di importanza.

### 8.2.2.2 Ticket di realizzazione e controllo

Gestiscono le due fasi di produzione di un documento: la realizzazione, effettuata dal redattore del documento, e il controllo, effettuato da un verificatore diverso dal redattore. Un ticket di controllo non può essere affidato ad un membro del team che ha partecipato alla redazione dello scritto da sottoporre in analisi, ovvero che ha precedentemente ricevuto un ticket di creazione sullo stesso documento.

### 8.2.2.3 Ticket di verifica

Rappresentano segnalazioni di errori, problematiche o incorrettezze riscontrate dai Verificatori durante le fase di verifica. Dovranno essere assegnati ad un membro del team che in quel momento ricopre un ruolo adatto ad intervenire per la correzione e in seguito riferificati da un *Verificatore*.

I ticket di verifica avranno specifici stadi di avanzamento:

* *Enqueued*: Il ticket è stato creato ed è in attesa di essere assegnato ad un X per la risoluzione del problema
* *In Progress*: Il ticket è stato assegnato ed è in lavorazione
* *Closed*: La lavorazione è terminata e il problema segnalato dal ticket è stato risolto
* *Approved* o *Rejected*: Un verificatore approva o rigetta il lavoro effettuato. Nel caso di rigetto deve venire creato un nuovo ticket per risolvere il problema.

### 8.2.2.4 Ticket di modifica

Rappresentano richieste di modifica fatte da membri del team al *Responsabile.* Quest’ultimo poi decide se accettare la richiesta di modifica o rifiutarla. Se la richiesta viene accettata, il *Responsabile* convoca una riunione per la discussione della modifica. Se la modifica viene rettificata dall’assemblea, le viene assegnata una prorità e viene messa nell’elenco di modifiche in attesa di implementazione.

Le priorità che possono essere assegnate sono le seguenti:

* Urgente: Da applicare il prima possibile
* Media: Da applicare entro la prossima milestone
* Bassa: L’implementazione non è obbligatoria, non c’è una scadenza definita.

# 9. Glossario

Ogni qualvolta, durante la stesura di un documento, si incorra in un termine tecnico o non di uso comune, esso dovrà essere inserito nel glossario insieme ad una sua precisa definizione. Se la definizione conterrà a sua volta termini di questo tipo, anche essi a loro volta dovranno essere inseriti nel glossario.

Le definizioni dei termini devono essere quanto più semplici e chiare possibile, e devono cercare di evitare ricorsioni o termini da glossario.

I termini nel glossario devono necessariamente essere elencati in ordine alfabetico.

Ogni definizione del termine del glossario deve contenere una spiegazione concisa; e’ preferibile che la definizione inizi con un sostantivo, in ogni caso non deve mai iniziare con l’espressione “e’ un”.

All’interno della definizione del termine, l’occorrenza del termine stesso non dovra’ contenere la notazione *termineG*.

10. Strumenti

Le risorse software che si utilizzeranno durante il processo di veriﬁca sono:

* **Correttore automatico TeXstudio**: per la scrittura di documenti è consigliato utilizzare l’ambiente graﬁco TeX-studio. Tale strumento integra i dizionari di OpenOﬃce.org e segnala i potenziali errori ortograﬁci presenti nel testo;
* **Aspell**: strumento per la correzione tipograﬁca dei documenti redatti in LATEX;
* **Glossario-script**: script, scritto dai componenti del gruppo, che marca i termini nel glossario con la simbologia sopracitata.

## 

12 Metodi

Vengono qui esplicitate le procedure con cui si eseguono l’analisi statica e dinamica per la veriﬁca dei documenti.

12.1 Analisi dei documenti

1. **Controllo sintattico e del periodo**

Utilizzando TeXstudio e GNU Aspell vengono evidenziati e corretti gli errori di

grammatica più evidenti. Ciascun documento dovrà essere sottoposto ad un walkthrough da parte dei veriﬁcatori per individuare errori di sintassi e periodi di difficile comprensione;

**2.**

## 12.Qualcosa Walkthrough

Questa tecnica consiste nella lettura totale dell'oggetto di verifica, senza presupposti, mirata a rilevare anomalie di cui non si e a conoscenza. E’ una tecnica usata principalmente nelle prime fasi del progetto in quanto i membri del gruppo non possiedono ancora l’esperienza necessaria per una verifica piu’ mirata. Verra’ utilizzata dal gruppo in prima persona durante la redazione dei documenti, anche da componenti del gruppo diversi dal redattore, data la natura ancora informale del documento analizzato. Inoltre essa verra implicitamente utilizzata da tutti gli strumenti software di analisi statica utilizzati per la verifica.

La verifica tramite Walkthrough viene effettuata in 4 fasi distinte:

1. *pianificazione* della verifica;
2. *lettura*;
3. *discussione*, fase in cui si riportano gli errori riscontrati, al fine di rendere piu’ specifica la ricerca;
4. *correzione* dei difetti.

## 12.Qualcosa+1 Inspection

Questa tecnica consiste nella ricerca di anomalie velocizzata da assunzioni riguardanti la tipologia e la posizione sulla base di conoscenze di tipo statistico, apprese dai verificatori attraverso la tecnica del walkthrough. Grazie a queste assunzioni l’inspection e’ una strategia piu’ rapida del walkthrough e non necessita delle lettura integrale del documento in oggetto.

Ogni applicazione della tecnica di inspection si divide in 4 fasi distinte:

1. pianificazione della verifica;
2. definizione della lista di controllo, ovvero il rierimento ad un preciso elenco di verifica mirata;
3. lettura;
4. correzione.

Durante l’applicazione del walkthrough ai documenti, sono state riportate le tipologie di errori più frequenti. La lista di controllo risultante è la seguente:

* Norme stilistiche:
  + Elenco puntato: non inizia con la lettera maiuscola;
  + Elenco puntato: non termina con il punto e virgola oppure con il punto se è l’ultimo elemento;
  + I ruoli di progetto: non scritti in corsivo.
* Latex:
  + Lettere accentate nelle variabili: non viene utilizzato il comando apposito;
  + Carattere di spaziatura: non deve essere utilizzato all’interno dei tag;
  + Macro LATEX: non viene scritta usando l’apposito comando \LaTeX{}.
* UML:
  + Il sistema non deve mai essere un attore;
  + Controllo ortograﬁco: deve essere eﬀettuato in modo dettagliato a causa dell’impossibilità di automatizzare i controlli sui diagrammi;
  + Direzione delle frecce non corrette;
  + Consistenza della nomenclatura tra i diagrammi e le descrizioni testuali nei documenti.

Per quanto riguarda la verifica mirata del codice, non abbiamo sviluppato un elenco di errori comuni, in quanto non compete alla fase di Analisi dei Requisiti la stesura di codice.

Qundi ci appoggiamo su tecniche di *best practiceG* assodate per avere un insieme di ispezioni di riferimento. Queste tecniche comprendono il controllo in piu’ campi:

* Errori nei dati:
  + le variabiliG sono state inizializzate prima di essere utilizzate?
  + le costanti sono state specificate e nominate correttamente?
  + gli *upper boundG ,* nello scorrimento di un arrayG, sono corretti?
  + esiste la possibilita’ del verificarsi di un overflowG?
* Errori nei controlli:
  + le condizioni negli statementG condizionali sono effettivamente corrette?
  + ogni ciclo e’ certo di terminare?
* Errori input/output:
  + le variabili di input sono state interamente utilizzate?
  + e’ stato assegnato un valore ad ogni variabile di output prima di essere esportate?
  + input imprevisti possono causare corruzione?
* Errori di interfaccia:
  + tutte le funzioni e le chiamate di metodi hanno il corretto numero di parametri?
  + i tipi dei parametri formali e attuali sono gli stessi?
  + tutti i parametri sono nell’ordine giusto?
  + se i componenti condividono della memoria, condividono anche la struttura di memorizzazione condivisa?
* Errori di gestione della memoria:
  + se una struttura a riferimenti viene modificata, tutti i riferimenti sono stati correttamente riassegnati?
  + se viene usata un’allocazione dinamica della memoria, lo spazio viene allocato correttamente?
  + lo spazio occupato viene esplicitamente deallocato, una volta che non e’ piu’ richiesto?
* Errori di gestione delle eccezioni:
  + tutte le possibili eccezioni vengono tracciate e gestite?