

# NORRIS Framework



FLAMETECH Inc.

## Norme di Progetto

### Informazioni sul documento

<b>Versione</b>	3.0.0
<b>Redazione</b>	Zanetti Davide Persegona Mattia
<b>Verifica</b>	Faggin Andrea
<b>Responsabile</b>	Cardin Andrea
<b>Uso</b>	Interno
<b>Lista di distribuzione</b>	<b>FlameTech Inc.</b> Prof. Vardanega Tullio Prof. Cardin Riccardo

### Descrizione

Regole adottate dal gruppo **FlameTech Inc.** durante tutto lo svolgimento del progetto

<b>Stato</b>	<b>Modifica</b>	<b>Autore</b>	<b>Ruolo</b>	<b>Data</b>	<b>Versione</b>
Approvato	Documento approvato	Cardin Andrea	Responsabile	2015/05/14	3.0.0
Verificato	Documento verificato	Faggin Andrea	Verificatore	2015/05/14	2.1.0
In Lavorazione	Corretto titolo paragrafo 2.1, spostata la sezione Gestione di Progetto nella sezione 3 - Processi Organizzativi	Zanetti Davide	Amministratore	2015/05/08	2.0.4
In Lavorazione	Rimossa sezione Glossario ed inserita come sottosezione del paragrafo 4.1	Persegona Mattia	Amministratore	2015/05/07	2.0.3
In Lavorazione	Modificata struttura della sezione 4 per separare maggiormente processi, strumenti e attività	Zanetti Davide	Amministratore	2015/05/07	2.0.2
In Lavorazione	Stesura sezione 2.1.3 per le norme sulla Codifica	Zanetti Davide	Amministratore	2015/04/28	2.0.1
Approvato	Documento approvato	Merlo Gianluca	Responsabile	2015/03/11	2.0.0
Verificato	Documento verificato	Faggin Andrea	Verificatore	2015/03/10	1.5.0
In Lavorazione	Stesura norme per Codifica	Meneguzzo Francesco	Amministratore	2015/02/25	1.4.3
In Lavorazione	Stesura norme per Progettazione	Sartor Michele	Amministratore	2015/02/24	1.4.2
In Lavorazione	Applicazione Correzioni RR - riorganizzazione contenuti	Sartor Michele	Amministratore	2015/02/23	1.4.1
Approvato	Documento approvato	Sartor Michele	Responsabile	2014/12/01	1.4.0
Verificato	Documento verificato	Merlo Gianluca	Verificatore	2014/12/01	1.3.0
In Lavorazione	Stesura sezione documenti	Faggin Andrea	Amministratore	2014/11/28	1.2.1
Verificato	Documento verificato	Merlo Gianluca	Verificatore	2014/11/28	1.2.0

<b>Stato</b>	<b>Modifica</b>	<b>Autore</b>	<b>Ruolo</b>	<b>Data</b>	<b>Versione</b>
In Lavorazione	Stesura Ambiente di lavoro, Processi Organizzativi e Riunioni	Faggin Andrea	Amministratore	2014/11/27	1.0.2
In Lavorazione	Stesura sezione comunicazioni	Faggin Andrea	Amministratore	2014/11/26	1.0.1
In Lavorazione	Creato scheletro del documento, iniziata stesura	Faggin Andrea	Amministratore	2014/11/26	1.0.0

## Indice

<b>1</b>	<b>Introduzione</b>	<b>1</b>
1.1	Scopo del documento . . . . .	1
1.2	Scopo del prodotto . . . . .	1
1.3	Glossario . . . . .	1
1.4	Riferimenti . . . . .	1
1.4.1	Normativi . . . . .	1
1.4.2	Informativi . . . . .	1
<b>2</b>	<b>Processi Primari</b>	<b>2</b>
2.1	Processo di Sviluppo . . . . .	2
2.1.1	Analisi dei Requisiti . . . . .	2
2.1.1.1	Classificazione dei requisiti . . . . .	2
2.1.1.2	<i>Casi d'uso<sub>G</sub></i> . . . . .	3
2.1.1.2.1	Software creazione diagrammi UML . . . . .	3
2.1.1.3	Tracciamento dei Requisiti e dei <i>Casi d'Uso<sub>G</sub></i> . . . . .	3
2.1.1.3.1	Software per il tracciamento . . . . .	3
2.1.2	Progettazione . . . . .	4
2.1.2.1	Specifica Tecnica . . . . .	4
2.1.2.1.1	Diagrammi <i>UML<sub>G</sub></i> . . . . .	4
2.1.2.1.2	<i>Design Pattern<sub>G</sub></i> . . . . .	4
2.1.2.1.3	Tracciamento delle componenti . . . . .	4
2.1.2.1.4	Test di integrazione . . . . .	5
2.1.2.2	Definizione di prodotto . . . . .	5
2.1.2.2.1	Diagrammi <i>UML<sub>G</sub></i> . . . . .	5
2.1.2.2.2	Definizione delle classi . . . . .	5
2.1.2.2.3	Tracciamento delle classi . . . . .	5
2.1.3	Codifica . . . . .	5
2.1.3.1	Convenzioni per la codifica . . . . .	5
2.1.3.1.1	Nomi . . . . .	6
2.1.3.1.2	Intestazione del codice . . . . .	6
2.1.3.1.3	Indentazione . . . . .	6
2.1.3.1.4	Uso della ricorsione . . . . .	6
2.1.3.2	Strumenti per la codifica . . . . .	6
<b>3</b>	<b>Processi Organizzativi</b>	<b>7</b>
3.1	Gestione di Progetto . . . . .	7
3.1.1	Pianificazione e controllo delle attività . . . . .	7
3.1.1.1	Software per la pianificazione . . . . .	7
3.1.2	Controllo e gestione delle risorse . . . . .	7
3.1.3	Controllo e gestione dei rischi . . . . .	7
3.1.3.1	Analisi dei rischi e Studio di Fattibilità . . . . .	8
3.2	Organizzazione interna . . . . .	8
3.3	Rapporti con il cliente . . . . .	10
3.3.1	Composizione e-mail . . . . .	10
3.3.1.1	Mittente . . . . .	10
3.3.1.2	Destinatario . . . . .	10
3.3.1.3	Oggetto . . . . .	10
3.3.1.4	Corpo dell'e-mail . . . . .	11

3.3.1.5	Allegati . . . . .	11
3.4	Comunicazioni interne . . . . .	11
3.5	Riunioni . . . . .	11
3.5.1	Frequenza delle riunioni . . . . .	11
3.5.2	Convocazione della riunione . . . . .	11
3.5.3	Svolgimento della riunione . . . . .	12
3.5.4	Verbale . . . . .	12
3.5.4.1	Riunione Interna . . . . .	12
3.5.4.2	Riunione Esterna . . . . .	12
3.6	Ambiente operativo . . . . .	12
3.6.1	Sistema operativo . . . . .	12
3.6.2	Software di presentazione . . . . .	12
3.7	Configuration Management . . . . .	12
3.7.1	<i>Repository<sub>G</sub></i> . . . . .	12
3.7.1.1	Struttura del <i>repository<sub>G</sub></i> . . . . .	12
3.7.1.2	Norme sui contenuti del <i>repository<sub>G</sub></i> . . . . .	13
3.7.1.2.1	Tipo di file . . . . .	13
3.7.1.3	Norme sui <i>commit<sub>G</sub></i> . . . . .	13
3.7.2	Software per il versionamento . . . . .	13
3.8	<i>Ticketing<sub>G</sub></i> . . . . .	14
3.8.1	Gestione dei <i>ticket<sub>G</sub></i> . . . . .	14
3.8.1.1	Creazione di un <i>ticket<sub>G</sub></i> . . . . .	15
3.8.2	Gestione delle <i>issue<sub>G</sub></i> . . . . .	15
3.8.2.1	Contenuto di una <i>issue<sub>G</sub></i> . . . . .	16
3.8.2.2	Creazione e completamento di <i>issue<sub>G</sub></i> . . . . .	16
<b>4</b>	<b>Processi di supporto</b>	<b>18</b>
4.1	Documentazione . . . . .	18
4.1.1	<i>Template<sub>G</sub></i> . . . . .	18
4.1.1.1	Struttura del <i>Template<sub>G</sub></i> . . . . .	18
4.1.1.1.1	Prima Pagina . . . . .	18
4.1.1.1.2	Diario delle modifiche . . . . .	19
4.1.1.1.3	Indici . . . . .	20
4.1.1.1.4	Struttura delle pagine . . . . .	20
4.1.2	Norme Tipografiche . . . . .	20
4.1.2.1	Stile del testo . . . . .	20
4.1.2.2	Formati . . . . .	21
4.1.2.3	Immagini . . . . .	21
4.1.2.4	Tabelle . . . . .	21
4.1.2.5	Elenchi . . . . .	22
4.1.2.6	Norme sui nomi dei file . . . . .	22
4.1.3	Strumenti per i documenti . . . . .	22
4.1.4	Ciclo di vita del documento . . . . .	23
4.1.5	Versionamento . . . . .	23
4.1.6	Classificazione dei documenti . . . . .	24
4.1.6.1	Documento informale . . . . .	24
4.1.6.2	Documento formale . . . . .	24
4.1.7	Glossario . . . . .	24
4.1.7.1	Inserimento del termine . . . . .	24

4.1.7.2	Ordine di apparizione . . . . .	24
4.2	Verifica . . . . .	25
4.2.1	Tecniche di Analisi . . . . .	25
4.2.1.1	Analisi Statica . . . . .	25
4.2.1.1.1	Inspection . . . . .	25
4.2.1.1.2	Walkthrough . . . . .	25
4.2.1.2	Analisi dinamica . . . . .	25
4.2.1.2.1	Test di unità . . . . .	26
4.2.1.2.2	Test di integrazione . . . . .	26
4.2.1.2.3	Test di regressione . . . . .	26
4.2.1.2.4	Test di sistema . . . . .	26
4.2.1.2.5	Test di accettazione . . . . .	26
4.2.2	Procedure . . . . .	27
4.2.2.1	Metriche errori riscontrati e gestione cambiamenti . . . . .	27
4.2.2.2	Verifica dei processi . . . . .	28
4.2.2.3	Verifica dei documenti . . . . .	28
4.2.2.4	Verifica tracciamento dei requisiti . . . . .	30
4.2.2.5	Verifica <i>UML<sub>G</sub></i> . . . . .	30
4.2.2.6	Verifica progettazione architetturale . . . . .	31
4.2.2.7	Verifica progettazione di dettaglio . . . . .	31
4.2.2.8	Verifica del codice . . . . .	31
4.2.3	Strumenti . . . . .	31
4.2.3.1	Strumenti per la verifica dei documenti . . . . .	31
4.2.3.2	Strumenti per la verifica del codice . . . . .	31
<b>A</b>	<b>Lista Controllo</b>	<b>33</b>



## Elenco delle tabelle

2	Gravità dell'errore e impatto su processi e prodotti . . . . .	27
3	Errori nei processi e modalità di risoluzione . . . . .	28
4	Errori nei documenti e modalità di risoluzione . . . . .	28



## Elenco delle figure

1	Pagina di FIRE . . . . .	4
2	Ripartizione ruoli Cardin . . . . .	9
3	Ripartizione ruoli Faggin . . . . .	9
4	Ripartizione ruoli Meneguzzo . . . . .	9
5	Ripartizione ruoli Merlo . . . . .	9
6	Ripartizione ruoli Persegona . . . . .	9
7	Ripartizione ruoli Sartor . . . . .	9
8	Ripartizione ruoli Zanetti . . . . .	10
9	Assegnazione di un <i>ticket<sub>G</sub></i> . . . . .	14
10	Server <i>Redmine<sub>G</sub></i> . . . . .	15
11	Esempio di <i>issue<sub>G</sub></i> . . . . .	16
12	Template di prima pagina . . . . .	19
13	Template del diario delle modifiche . . . . .	20
14	Procedura per l'analisi dei documenti . . . . .	29
15	Gestione delle anomalie . . . . .	30



## 1 Introduzione

### 1.1 Scopo del documento

Lo scopo del documento è definire le norme che i componenti del gruppo **FlameTech Inc.** adotteranno durante la realizzazione del progetto Norris.

Tutti i componenti del gruppo sono tenuti a leggere il documento ed applicare le norme descritte allo scopo di migliorare l'uniformità del materiale prodotto, l'efficienza del lavoro e di ridurre al minimo il numero di errori. Le norme descritte in questo documento riguardano:

- convenzioni da usare all'interno dei documenti;
- interazioni tra i componenti del gruppo;
- interazioni con i proponenti;
- modalità di lavoro;
- l'ambiente su cui verrà sviluppato il prodotto;

### 1.2 Scopo del prodotto

Lo scopo del prodotto è la realizzazione di un *framework<sub>G</sub>* per *Node.js<sub>G</sub>*, compatibile con l'utilizzo standard dei *middleware<sub>G</sub>* di *Express<sub>G</sub>* in versione 4.x, per la realizzazione rapida di *client<sub>G</sub>* web per la visualizzazione di grafici aggiornabili in tempo reale.

### 1.3 Glossario

Per evitare ogni possibile ambiguità che potrebbe sorgere verrà allegato il *Glossario\_ver4.0.0* dove verranno inseriti termini tecnici, acronimi, termini di dominio ed eventuali parole che potrebbero comportare delle incomprensioni e delle ambiguità nella lettura dei documenti. Per rendere la lettura più facile i termini verranno riportati in corsivo ed in pedice verrà posta una "G" maiuscola. (Esempio: *Android<sub>G</sub>*).

### 1.4 Riferimenti

#### 1.4.1 Normativi

- **Analisi dei Requisiti:** *AnalisiRequisiti\_ver4.0.0*;
- **Piano di Progetto:** *PianoDiProgetto\_ver4.0.0*;
- **Piano di Qualifica:** *PianoDiQualifica\_ver3.0.0*;
- **Studio di Fattibilità:** *StudioDiFattibilità\_ver1.4.0*.

#### 1.4.2 Informativi

- **Github<sub>G</sub>:** <http://www.github.com>;
- **Server Redmine<sub>G</sub>:** <http://188.166.60.146/redmine>.

## 2 Processi Primari

### 2.1 Processo di Sviluppo

#### 2.1.1 Analisi dei Requisiti

Oltre alla stesura dello *StudioDiFattibilità*, agli *Analisti* spetta il compito di realizzare l'*AnalisiRequisiti*. La stesura del documento dovrà rispettare i criteri che verranno ora descritti.

##### 2.1.1.1 Classificazione dei requisiti

Ogni requisito, individuati tramite un incontro con il Proponente o l'analisi del capitolato, dovrà essere identificato rispettando la seguente codifica:

R[importanza][tipo][codice identificativo]

- **Importanza:** può assumere solo uno dei seguenti valori:
  - **A:** requisito obbligatorio;
  - **B:** requisito desiderabile;
  - **C:** requisito opzionale.
- **Tipo:**
  - **F:** Requisito Funzionale;
  - **Q:** Requisito Qualitativo;
  - **P:** Requisito Prestazionale;
  - **V:** Requisito di Vincolo.
- **Codice identificativo:** è il codice univoco di ogni requisito e sarà espresso da un numero seguendo l'ordine gerarchico (esempio: 1.1.5).

Ogni requisito, oltre ad essere codificato, avrà una breve descrizione e la fonte da cui è stato estrapolato. Le uniche **fonti** possibili sono:

- **Capitolato:** il requisito è estrapolato direttamente dal capitolato;
- **Caso d'uso<sub>G</sub>:** questo requisito non è emerso direttamente dal capitolato ma durante la realizzazione dei *casi d'uso<sub>G</sub>*;
- **Padre:** nasce dalla necessità di suddividere un requisito di livello superiore;
- **Interno:** gli *Analisti* hanno ritenuto importante l'aggiunta di questo requisito;
- **Verbale:** il requisito è emerso durante l'incontro con il Committente o una riunione del gruppo.

### 2.1.1.2 *Casi d'uso<sub>G</sub>*

Concluso lo *Studio di Fattibilità* spetta sempre agli *Analisti* l'analisi dei *casi d'uso<sub>G</sub>*. Anche i *casi d'uso<sub>G</sub>* dovranno essere classificati secondo una codifica precisa. La codifica scelta è:

UC[codice identificativo]

- **Codice identificativo:** codice identificativo gerarchico di ogni caso d'uso. Il codice univoco sarà così strutturato :

[codice univoco del padre.codice univoco di livello].

Nel caso di derivazioni superiori a due livelli il codice corrisponderà alla sequenza dei codici di derivazione di tutti i livelli.

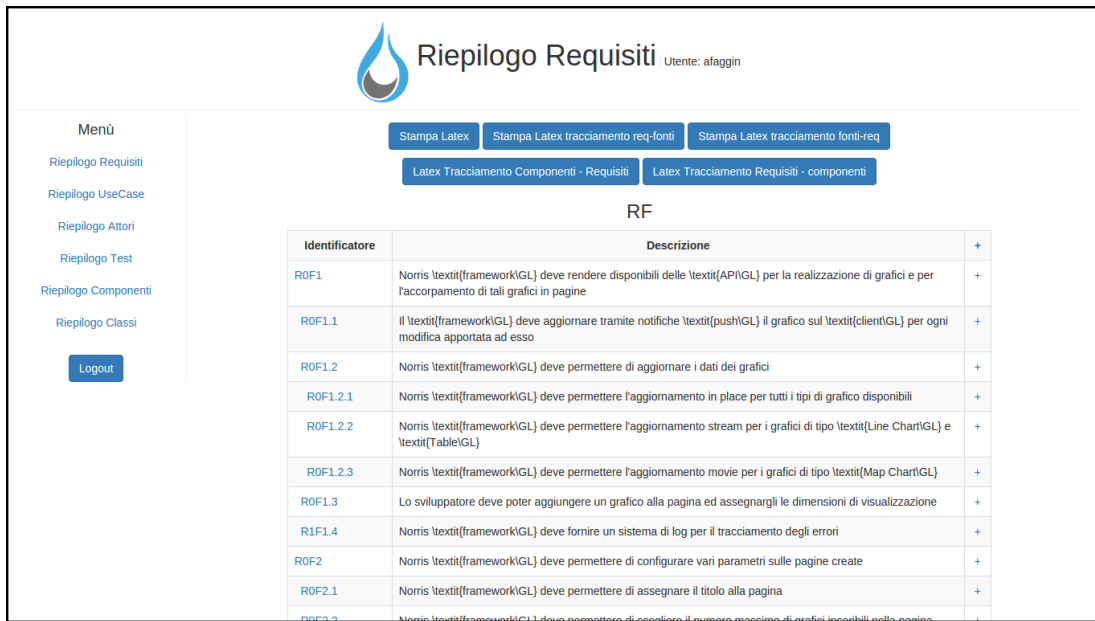
**2.1.1.2.1 Software creazione diagrammi UML** Il software utilizzato per la creazione dei diagrammi *UML<sub>G</sub>* nella versione 2.0 è *Astah<sub>G</sub> Community Edition*. Con l'utilizzo del software verranno realizzati i diagrammi dei *casi d'uso<sub>G</sub>*, delle classi, dei *package<sub>G</sub>*, di sequenza e di attività.

### 2.1.1.3 Tracciamento dei Requisiti e dei *Casi d'Uso<sub>G</sub>*

Per automatizzare il tracciamento dei requisiti e dei *casi d'uso<sub>G</sub>* il team ha deciso di realizzare un applicativo, per uso interno, che si appoggia su un *database<sub>G</sub>*. In questo modo ad ogni modifica dei requisiti e/o dei *casi d'uso<sub>G</sub>* si avrà un aggiornamento automatico della corrispondenza con le fonti.

L'applicativo, oltre al tracciamento dei requisiti e dei *casi d'uso<sub>G</sub>*, è stato dotato di una apposita funzione che esporta direttamente in codice  $\text{\LaTeX}$  il contenuto del *database<sub>G</sub>*, in maniera da velocizzare la realizzazione del documento dell'*AnalisiRequisiti\_ver4.0.0*. Nonostante questo sia automatizzato spetta agli *Analisti* controllare la correttezza e la corrispondenza biunivoca di un requisito con una o più fonti. Nel caso dovessero essere riscontrate delle anomalie nel funzionamento bisogna informare il *Responsabile*, che dovrà prendere le decisioni su come procedere per la correzione del problema.

**2.1.1.3.1 Software per il tracciamento** Per realizzare il tracciamento dei requisiti, dei *casi d'uso<sub>G</sub>* e delle fonti il team ha realizzato un proprio software, denominato *FIRE (Flametech Internal Requirements Engine)*. Il software appoggia su un *database<sub>G</sub> MySQL<sub>G</sub>* ed è reperibile all'indirizzo <http://188.166.60.146/firev2/>. Per automatizzare la stesura del documento *Analisi dei Requisiti* è stata realizzata una apposita funzione che esporta il contenuto in codice  $\text{\LaTeX}$ .



Identificatore	Descrizione	
R0F1	Norris <code>\textit{frameworkGL}</code> deve rendere disponibili delle <code>\textit{APIGL}</code> per la realizzazione di grafici e per l'accorpamento di tali grafici in pagine	+
R0F1.1	Il <code>\textit{frameworkGL}</code> deve aggiornare tramite notifiche <code>\textit{pushGL}</code> il grafico sul <code>\textit{clientGL}</code> per ogni modifica apportata ad esso	+
R0F1.2	Norris <code>\textit{frameworkGL}</code> deve permettere di aggiornare i dati dei grafici	+
R0F1.2.1	Norris <code>\textit{frameworkGL}</code> deve permettere l'aggiornamento in place per tutti i tipi di grafico disponibili	+
R0F1.2.2	Norris <code>\textit{frameworkGL}</code> deve permettere l'aggiornamento stream per i grafici di tipo <code>\textit{Line ChartGL}</code> e <code>\textit{TableGL}</code>	+
R0F1.2.3	Norris <code>\textit{frameworkGL}</code> deve permettere l'aggiornamento movie per i grafici di tipo <code>\textit{Map ChartGL}</code>	+
R0F1.3	Lo sviluppatore deve poter aggiungere un grafico alla pagina ed assegnargli le dimensioni di visualizzazione	+
R1F1.4	Norris <code>\textit{frameworkGL}</code> deve fornire un sistema di log per il tracciamento degli errori	+
R0F2	Norris <code>\textit{frameworkGL}</code> deve permettere di configurare vari parametri sulle pagine create	+
R0F2.1	Norris <code>\textit{frameworkGL}</code> deve permettere di assegnare il titolo alla pagina	+
R0F2.2	Norris <code>\textit{frameworkGL}</code> deve permettere di scegliere il numero massimo di grafici inseribili nella pagina	+

Figura 1: Pagina di FIRE

## 2.1.2 Progettazione

### 2.1.2.1 Specifica Tecnica

I *Progettisti* nella *Specifica Tecnica* devono specificare la struttura ad alto livello dell'architettura che intendono implementare nel prodotto e nei singoli componenti. Oltre a questo devono provvedere alla progettazione dei test di integrazione che dovranno essere effettuati.

**2.1.2.1.1 Diagrammi  $UML_G$**  Nella fase di progettazione il gruppo farà uso dei seguenti diagrammi:

- Diagrammi delle classi;
- Diagrammi di sequenza;
- Diagrammi delle attività;
- Diagrammi dei *package<sub>G</sub>*.

**2.1.2.1.2 *Design Pattern<sub>G</sub>*** La scelta dei *Design Pattern<sub>G</sub>* da utilizzare per l'architettura del prodotto spetta ai *Progettisti*. Per ogni *design pattern<sub>G</sub>* adottato dovranno essere forniti una breve descrizione ed un diagramma della struttura.

**2.1.2.1.3 Tracciamento delle componenti** Le componenti individuate nella *Specifica Tecnica<sub>ver2.0.0</sub>* verranno tracciate ai requisiti funzionali e qualitativi individuati e descritti nell'*AnalisiRequisiti<sub>ver4.0.0</sub>* che vengono soddisfatti dalle stesse. Il tracciamento sarà realizzato in maniera automatica dal software *FIRE<sub>G</sub>*, così facendo sarà possibile avere continuamente sotto controllo lo stato di avanzamento dell'attività di progettazione ed inoltre sarà possibile monitorare la copertura dei requisiti.

**2.1.2.1.4 Test di integrazione** Ai *Progettisti* spetta anche il compito di definire i test di integrazione che dovranno essere eseguiti per verificare che le varie componenti del sistema si integrino in maniera corretta.

#### 2.1.2.2 Definizione di prodotto

Nella *Definizione di Prodotto* i *Progettisti* devono descrivere la progettazione di dettaglio del sistema, e ampliare quanto descritto nella *Specifica Tecnica*, specificandone i particolari. In questo documento devono essere definiti in maniera dettagliata tutte le singole unità che andranno a costituire il prodotto finale, semplificando l'attività di codifica. Saranno inoltre imposti dei limiti al *Programmatore* che non potrà così avere libertà di realizzazione. Oltre a questo saranno progettati anche i test per ogni singola unità, descritti nel documento *PianoDiQualifica\_ver3.0.0*.

**2.1.2.2.1 Diagrammi UML<sub>G</sub>** I seguenti diagrammi dovranno essere affinati e aggiornati:

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

**2.1.2.2.2 Definizione delle classi** All'interno della *Definizione di Prodotto* dovrà essere definita ogni classe che si intende utilizzare, e per ognuna di esse dovranno essere definiti:

- L'elenco dei metodi;
- L'elenco degli attributi;
- Le funzionalità modellate;
- La motivazione per cui la classe è necessaria.

Il software *FIRE<sub>G</sub>* genererà automaticamente il codice  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$  contenente la descrizione della classe da includere nel documento.

**2.1.2.2.3 Tracciamento delle classi** Le classi individuate dovranno, come le componenti, essere tracciate ai requisiti in *AnalisiRequisiti\_ver4.0.0* che soddisfano. Il tracciamento sarà realizzato in maniera automatica dal software *FIRE<sub>G</sub>*, così facendo sarà possibile avere continuamente sotto controllo lo stato di avanzamento dell'attività di progettazione ed inoltre sarà possibile monitorare la copertura dei requisiti rispetto alle classi.

### 2.1.3 Codifica

#### 2.1.3.1 Convenzioni per la codifica

Tutti i file realizzati dovranno essere in codifica *UTF-8<sub>G</sub>* senza l'utilizzo del *BOM<sub>G</sub>*. Per andare a capo invece verrà usato *LF<sub>G</sub>(U+000A)*. Agli sviluppatori è richiesto di attenersi alle *Javascript Coding Conventions<sub>G</sub>*.



Qualora qualche sviluppatore volesse modificare le convenzioni sopra descritte deve inviarne formale richiesta al *Responsabile*, il quale dovrà valutarne la proposta e in caso la richiesta fosse accettata deve comunicarla a tutti gli altri sviluppatori per uniformarsi.

**2.1.3.1.1 Nomi** Per i nomi delle variabili, nomi delle classi e delle funzioni dovrà essere usata esclusivamente la lingua inglese e si dovrà utilizzare la notazione *CamelCase<sub>G</sub>*. Le classi dovranno avere la prima lettera maiuscola, mentre le funzioni e le variabili dovranno averla minuscola.

**2.1.3.1.2 Intestazione del codice** Tutti i file che al loro interno conterranno del codice dovranno contenere la seguente intestazione:

```
/**
 * Name : index.js
 * Module : Back-end
 * Location : /
 *
 * History :
 *
 * Version          Date          Programmer
 * =====
 * 0.0.1            2015/04/05      Faggin Andrea
 * -----
 * ...
 * =====
 */
```

**2.1.3.1.3 Indentazione** L'indentazione del codice deve rispettare quanto descritto all'indirizzo <https://docs.npmjs.com/misc/coding-style#indentation>, inoltre per la notazione oggetto di *JavaScript<sub>G</sub>* verrà rispettata la specifica descritta allo stesso indirizzo.

**2.1.3.1.4 Uso della ricorsione** L'utilizzo della ricorsione in fase di programmazione deve essere, se possibile, totalmente evitata. Nel caso venisse utilizzata è obbligatorio fornire dei test che ne dimostrino l'effettiva terminazione e l'impatto che ha nell'occupazione della memoria. Al termine di questi test verrà valutato se può essere mantenuta o se deve essere trasformata nella corrispettiva versione iterativa.

## 2.1.3.2 Strumenti per la codifica

Per la stesura del codice ogni *Programmatore* dovrà utilizzare l'*IDE<sub>G</sub>* Eclipse. Sarà fatto uso della versione Java EE, appositamente creata per la scrittura di codice *JavaScript<sub>G</sub>*. Lo strumento è scaricabile gratuitamente dal sito <https://www.eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/lunasr2>.

## 3 Processi Organizzativi

### 3.1 Gestione di Progetto

La gestione dell'intero progetto è demandata al *Responsabile*, che dovrà garantire che le attività vengano svolte in maniera corretta. Per farlo si potrà avvalere di strumenti che gli permettano di:

- pianificare e controllare le attività;
- gestire e controllare le risorse;
- analizzare e gestire i rischi.

#### 3.1.1 Pianificazione e controllo delle attività

Per eseguire e garantire una buona pianificazione, il *Responsabile* dovrà realizzare, tramite il software *ProjectLibre<sub>G</sub>*, il diagramma di *Gantt<sub>G</sub>* per tutte le attività presenti nel *PianoDiProgetto\_ver4.0.0*. Fatta la pianificazione dovrà assegnare ogni attività ad un componente del gruppo e accertarsi che stiano avanzando in maniera corretta.

##### 3.1.1.1 Software per la pianificazione

Per organizzare la pianificazione del lavoro e delle risorse è stato scelto di utilizzare *ProjectLibre<sub>G</sub>*. La scelta di utilizzare questo software è principalmente dovuta al sistema operativo su cui si è scelto di lavorare. Un'altra valida alternativa sarebbe stata *Microsoft Project<sub>G</sub>*, che però è utilizzabile solamente su piattaforma Microsoft Windows. I diagrammi prodotti con *ProjectLibre* sono compatibili con le versioni 2010 e successive di *Microsoft Project<sub>G</sub>*. Il software permette di:

- realizzare i diagrammi di *Gantt<sub>G</sub>*;
- assegnare le risorse alle attività;
- creare automaticamente sia i diagrammi di *PERT<sub>G</sub>* sia la *Work Breakdown Structure (WBS)*;
- calcolare la *Schedule Variance (SV)<sub>G</sub>* e il *Budget Variance (BV)<sub>G</sub>*.

#### 3.1.2 Controllo e gestione delle risorse

Oltre al controllo delle attività il *Responsabile* dovrà monitorare le varie risorse e verificare l'avanzamento di ogni processo, come descritto nel *PianoDiProgetto\_ver4.0.0*.

#### 3.1.3 Controllo e gestione dei rischi

Per tutta la durata del progetto il *Responsabile* dovrà monitorare i possibili rischi, sia quelli già descritti nel *PianoDiProgetto\_ver4.0.0* sia quelli nuovi che potrebbero emergere. Per mitigare gli effetti collaterali dovrà attuare gli accorgimenti previsti nel *PianoDiProgetto\_ver4.0.0*.

### 3.1.3.1 Analisi dei rischi e Studio di Fattibilità

In seguito alla presentazione dei capitolati, il *Responsabile di Progetto* deve convocare delle riunioni che serviranno al confronto tra i membri del team per confrontarsi su quale capitolato svolgere e sulla possibilità di realizzazione entro i termini previsti. Da queste riunioni gli *Analisti* usciranno con la conoscenza di quale sia la preferenza e la preparazioni, in merito ad ogni capitolato, dei rispettivi membri del gruppo. Da questo momento è compito degli *Analisti* realizzare lo *StudioDiFattibilità* tenendo conto dei vari aspetti:

- **Rapporto Costi/Benefici:** gli *Analisti* dovranno conoscere quali sono i competitor ed i prodotti simili, il costo di realizzazione del progetto e quali devono essere i requisiti obbligatori che dovranno essere soddisfatti;
- **Dominio Applicativo e Tecnologico:** conoscenza di quali saranno le tecnologie da applicare, problematiche emerse dal capitolato e conoscenza del dominio applicativo;
- **Rischi:** conoscenza di quali potrebbero essere i punti critici durante la realizzazione, difficoltà nell'analisi, individuazione dei requisiti e come questi dovranno poi essere verificati. Gli *Analisti* dovranno inoltre conoscere le lacune di ogni membro del gruppo per quanto riguarda l'ambito tecnico e applicativo.

Al termine della realizzazione dello *StudioDiFattibilità\_ver1.4.0* sarà indetta una riunione per presentare la decisione finale sul capitolato.

## 3.2 Organizzazione interna

Durante la realizzazione del progetto i membri del gruppo andranno a ricoprire i ruoli solitamente presenti all'interno di un'azienda. Ciascun componente del gruppo dovrà ricoprire almeno una volta ogni ruolo. Per garantire una coerente rotazione dei ruoli, con lo scopo di non creare conflitti è necessario che le attività di stesura e verifica vengano pianificate dettagliatamente e che i soggetti interessati rispettino i compiti a loro assegnati.

I ruoli verranno riassegnati all'inizio di ogni fase interna del progetto, come pianificato nel documento *PianoDiProgetto\_ver4.0.0*, ad eccezione della fase **B**, che contiene un numero ridotto di attività, peraltro strettamente collegate alla fase precedente. La pianificazione è stata effettuata tenendo conto delle necessità di ciascun membro, e ha portato alle seguenti ripartizioni.

N.B. Vengono utilizzate delle abbreviazioni per indicare i ruoli del progetto didattico, conformi a quelle utilizzate in *PianoDiProgetto\_ver4.0.0*

- **RE:** Identifica il ruolo di *Responsabile*.
- **AN:** Identifica il ruolo di *Analista*.
- **PG:** Identifica il ruolo di *Progettista*.
- **AM:** Identifica il ruolo di *Amministratore*.
- **PR:** Identifica il ruolo di *Programmatore*.
- **VR:** Identifica il ruolo di *Verificatore*.



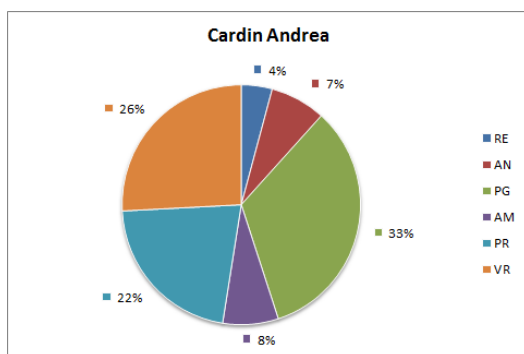


Figura 2: Ripartizione ruoli Cardin

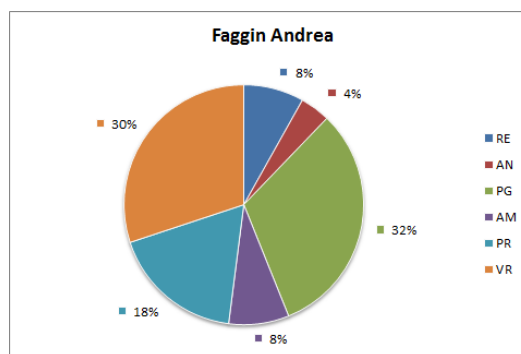


Figura 3: Ripartizione ruoli Faggin

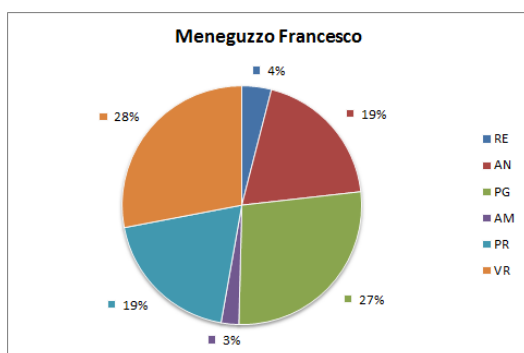


Figura 4: Ripartizione ruoli Meneguzzo

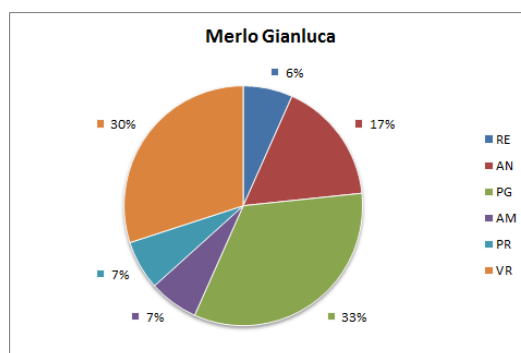


Figura 5: Ripartizione ruoli Merlo

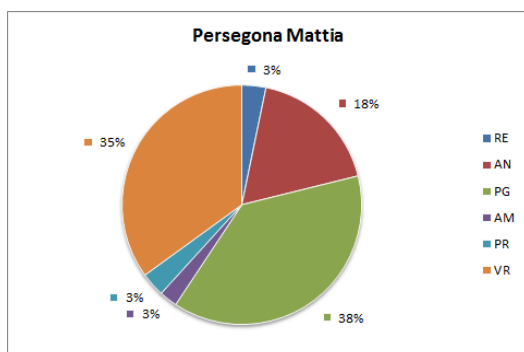


Figura 6: Ripartizione ruoli Persegona

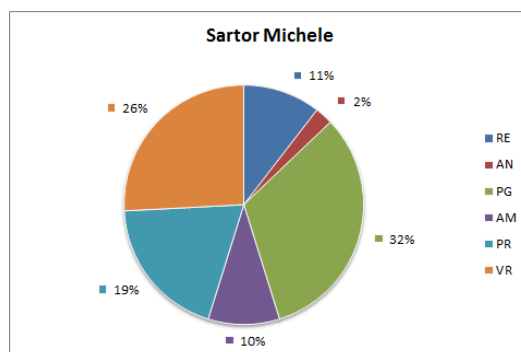


Figura 7: Ripartizione ruoli Sartor

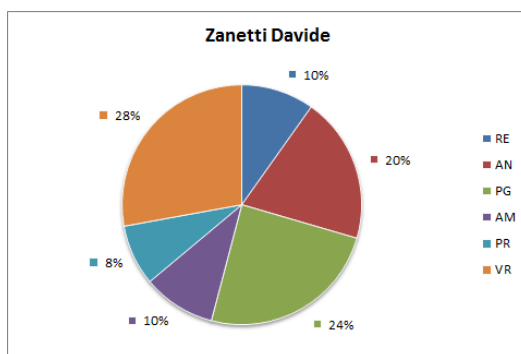


Figura 8: Ripartizione ruoli Zanetti

Sarà poi compito del *Verificatore* controllare attentamente il diario delle modifiche di ogni documento per individuare eventuali incongruenze. Si descrivono ora i diversi ruoli di progetto, con le relative responsabilità e le modalità operative affinché essi possano svolgere i compiti assegnati con l'ausilio dei software scelti per il progetto.

### 3.3 Rapporti con il cliente

Il gruppo **FlameTech Inc.** comunicherà con il Committente e il Proponente tramite un apposito indirizzo e-mail:

flametech.inc@gmail.com

L'e-mail sarà l'unico modo con cui il gruppo comunicherà con l'esterno. L'unica persona ad avere accesso all'indirizzo e-mail sarà il *Responsabile*. Spetterà poi sempre allo stesso *Responsabile* il compito di comunicare agli altri componenti del team il contenuto delle discussioni, ed eventualmente inoltrarlo.

#### 3.3.1 Composizione e-mail

Per tutte le e-mail si è deciso di fare riferimento alla struttura riportata di seguito:

##### 3.3.1.1 Mittente

Riguardo alle e-mail indirizzate all'esterno del gruppo, per esempio al Committente o Proponente, deve essere usata obbligatoriamente l'e-mail flametech.inc@gmail.com, che può essere inviata solamente dal *Responsabile*. Per le e-mail interne ogni componente userà il proprio indirizzo.

##### 3.3.1.2 Destinatario

Una e-mail indirizzata all'esterno del gruppo può essere al Proponente, al Prof. Vardanega Tullio o al Prof. Cardin Riccardo.

##### 3.3.1.3 Oggetto

L'oggetto di ogni e-mail deve essere breve, possibilmente non già usato in altre comunicazione ed il più esaustivo possibile. Per qualsiasi conversazione già avviata non è possibile modificarne l'oggetto.

#### 3.3.1.4 Corpo dell'e-mail

Nel corpo dell'e-mail bisogna essere il più esaustivi possibile ed inserire tutte le informazioni che sono necessarie per fare in modo che il messaggio risulti chiaro e facilmente comprensibile al destinatario. Nel caso si tratti di una risposta o di un inoltrò è obbligatorio aggiungere il contenuto in testa all' e-mail ed è consigliabile non cancellare il resto del testo così da avere sempre a portata di mano tutta la conversazione avvenuta fino a quel momento.

#### 3.3.1.5 Allegati

L'uso degli allegati è concesso ma deve essere usato il meno possibile e solamente nel caso ve ne sia necessità.

### 3.4 Comunicazioni interne

Le comunicazioni interne saranno tenute tramite un gruppo privato nel *social network* *Facebook<sub>G</sub>*. Tale gruppo sarà nominato [SWE] Norris, e sarà visibile ai soli membri del team. Per le comunicazioni veloci verrà utilizzata una chat di gruppo privata per i soli membri del team usando il software *Telegram<sub>G</sub>*. Tale chat sarà nominata FlameTech Inc.. In caso di decisioni prese tramite questo metodo di comunicazione il *Responsabile* dovrà notificare in modo permanente le scelte effettuate tramite un post permanente nel gruppo [SWE] Norris. Nel caso sia richiesto l'utilizzo di una chat o di una videoconferenza, verrà utilizzata l'applicazione *Hangouts<sub>G</sub>*.

### 3.5 Riunioni

#### 3.5.1 Frequenza delle riunioni

Il *Responsabile* ha stabilito che le riunioni interne saranno organizzate con una cadenza quindicinale. In casi di necessità o su richiesta da parte di alcuni membri del gruppo, previa conferma del *Responsabile*, potranno essere indette ulteriori riunioni. Diversamente le riunioni esterne verranno convocate dal *Responsabile* in accordo con il Committente, previa la disponibilità dello stesso, quando vi sarà la necessità di chiarimenti o di spiegazioni riguardo al progetto.

La data della riunione sarà comunicata dal *Responsabile* con almeno due giorni di anticipo in modo da permettere ai membri del gruppo di essere disponibili ed in caso di mancata disponibilità di raggiungere un accordo per un'occorrenza successiva.

#### 3.5.2 Convocazione della riunione

La convocazione di una riunione deve essere comunicata in via ufficiale dal *Responsabile* tramite il gruppo su *Facebook<sub>G</sub>* del team, per cui visibile a tutti i membri che dovranno confermare la presenza. La convocazione di una riunione potrà essere richiesta anche da uno dei componenti del team tramite l'invio di una comunicazione al *Responsabile*, il quale avrà poi il compito di decidere se accettare la proposta oppure di rigettarla, prendendo in considerazione il fatto segnalato per la successiva riunione periodica.

Le riunioni che verranno organizzate non comprenderanno necessariamente tutti i componenti del team ma potranno comprenderne solamente alcuni.

Ad ogni riunione è prevista la realizzazione di un verbale la cui condivisione fra i componenti del team permetterà di aggiornare anche gli assenti sulle decisioni prese e gli argomenti trattati. Vedi sez. 3.5.4.

### 3.5.3 Svolgimento della riunione

All'inizio di ogni riunione, sia questa interna oppure esterna, dovrà essere controllata la presenza di tutti i membri previsti e, fra di essi, dovrà essere scelto un segretario con il compito di annotare ogni argomento e domanda che viene posta in maniera tale da poter poi redigere in maniera consona il verbale della riunione.

Durante la riunione è chiesto a tutti i presenti di tenere un comportamento corretto ed adeguato al luogo in cui si trovano così da poter svolgere al meglio la riunione. Il segretario ha inoltre il compito di controllare che tutti i punti che erano stati comunicati all'ordine del giorno vengano discussi e non venga tralasciato nulla.

### 3.5.4 Verbale

#### 3.5.4.1 Riunione Interna

Il verbale di una riunione interna in cui prendono parte solo i componenti del gruppo sarà un documento informale che servirà a tenere traccia degli argomenti discussi durante la riunione. Il documento redatto verrà poi condiviso a tutti i componenti del gruppo **FlameTech Inc.** vedi sez. 4.1.6.1.

#### 3.5.4.2 Riunione Esterna

Il verbale di una riunione esterna sarà invece un documento formale ed ufficiale che potrà avere un valore normativo. Tale documento sarà caricato nel *repository<sub>G</sub>* all'interno della cartella denominata come **esterna**.

## 3.6 Ambiente operativo

Per coordinarsi nella realizzazione del prodotto e non incorrere in problematiche legate alla diversità dell'ambiente di lavoro il team ha scelto di adottare il medesimo ambiente.

### 3.6.1 Sistema operativo

Tutti i membri del gruppo operano su sistema operativo Linux, specificamente nella distribuzione *Kubuntu<sub>G</sub>* x64 con release 12.04 o successive.

### 3.6.2 Software di presentazione

Per la creazione delle presentazioni verrà usato *Prezi<sub>G</sub>*. La scelta è ricaduta su questo software perché è gratuito, online ed è possibile operarvi in più persone contemporaneamente.

## 3.7 Configuration Management

In questo capitolo sono descritte le attività e gli strumenti necessari per organizzare e gestire la configurazione del progetto.

### 3.7.1 *Repository<sub>G</sub>*

#### 3.7.1.1 Struttura del *repository<sub>G</sub>*

I membri del gruppo sono indotti a rispettare l'organizzazione dei file all'interno di tale struttura per permettere una facile ricerca e per mantenere un'adeguata organizzazione dei documenti. In particolare la *root<sub>G</sub>* conterrà solamente:

- **Documentazione:** una cartella contenente la documentazione di progetto a sua volta suddivisa in:
  - **Interna:** contenente la documentazione interna al gruppo e che non sarà distribuita esternamente;
  - **Esterna:** contenente i documenti realizzati e che saranno poi distribuiti esternamente al gruppo;
  - **Template:** contenente il *template<sub>G</sub>* utilizzato per la realizzazione dei documenti.
- **Codice:** in questa sezione troviamo i file contenenti il software prodotto.

### 3.7.1.2 Norme sui contenuti del *repository<sub>G</sub>*

Per poter essere aggiunto nel *repository<sub>G</sub>*, un elemento dovrà soddisfare vari requisiti.

**3.7.1.2.1 Tipo di file** Sono stati esclusi dal caricamento nel *repository<sub>G</sub>* gran parte dei file temporanei generati da *IDE<sub>G</sub>*, editor *L<sup>A</sup>T<sub>E</sub>X* e sistema operativo. I tipi degli elementi che possono essere aggiunti sono filtrati anche in base al percorso di caricamento, per garantire ordine nei contenuti in modo automatico.

### 3.7.1.3 Norme sui *commit<sub>G</sub>*

Ogni qualvolta si termina una modifica ad un file, sarà eseguito l'update nel *repository<sub>G</sub>* tramite il comando *commit<sub>G</sub>*. Il *commit<sub>G</sub>* verrà effettuato tramite il seguente comando:

```
git commit -m "motivo" nomefile
```

Il motivo del *commit<sub>G</sub>* deve essere strutturato in due parti. La motivazione viene inserita fra parentesi quadre per facilitare la comprensione dell'ambito della modifica; a riguardo si propongono tre categorie:

- **[STE]:** rappresenta la stesura del documento o di una sua parte;
- **[MOD]:** per indicare un aggiornamento per ampliare il documento;
- **[REV]:** sarà utilizzato per indicare che sono state effettuate modifiche in seguito a correzioni del documento.

In seguito si dovrà trovare una breve ma chiara descrizione delle operazioni effettuate. L'operazione di *commit<sub>G</sub>* deve essere eseguita solo al termine delle operazioni di modifica o quando ci sono modifiche sostanziali ai file.

## 3.7.2 Software per il versionamento

Per il versionamento il team ha deciso di usare *Git<sub>G</sub>*. La scelta è ricaduta su questo software perché era già conosciuto da tutti i componenti del team. In concomitanza alla scelta di utilizzare *Git<sub>G</sub>* è stato deciso di appoggiarsi a *GitHub<sub>G</sub>* per mantenere lo storico di tutte le modifiche apportate ai file e ai documenti relativi al progetto. La scelta è dovuta anche alla richiesta manifestata dal Proponente nel capitolato. *GitHub<sub>G</sub>* sarà sfruttato dal team per tenere il *repository<sub>G</sub>* del progetto.

### 3.8 *Ticketing<sub>G</sub>*

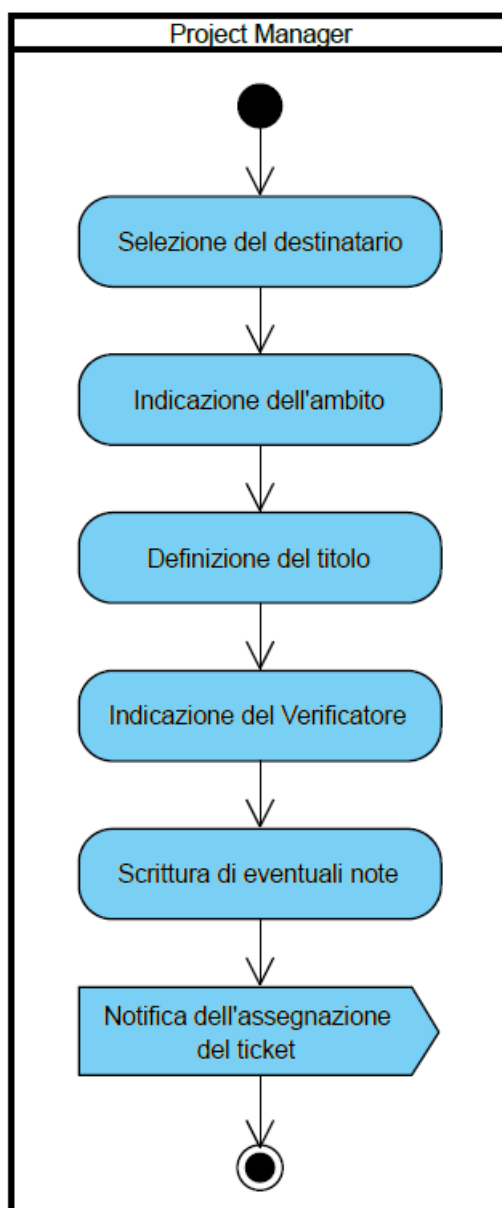


Figura 9: Assegnazione di un *ticket<sub>G</sub>*

#### 3.8.1 Gestione dei *ticket<sub>G</sub>*

Per la gestione del sistema di *ticketing<sub>G</sub>* del progetto è stato scelto di mettere in funzione un *server<sub>G</sub>* contenente *Redmine<sub>G</sub>*, all'indirizzo: <http://188.166.60.146/redmine>, su cui è stato creato il progetto. La piattaforma fornisce:

- il diagramma di *Gantt<sub>G</sub>* delle attività;
- l'assegnazione dei *ticket<sub>G</sub>*;
- un calendario per la visualizzazione delle scadenze assegnate;

- un sistema per il rendiconto del tempo speso.

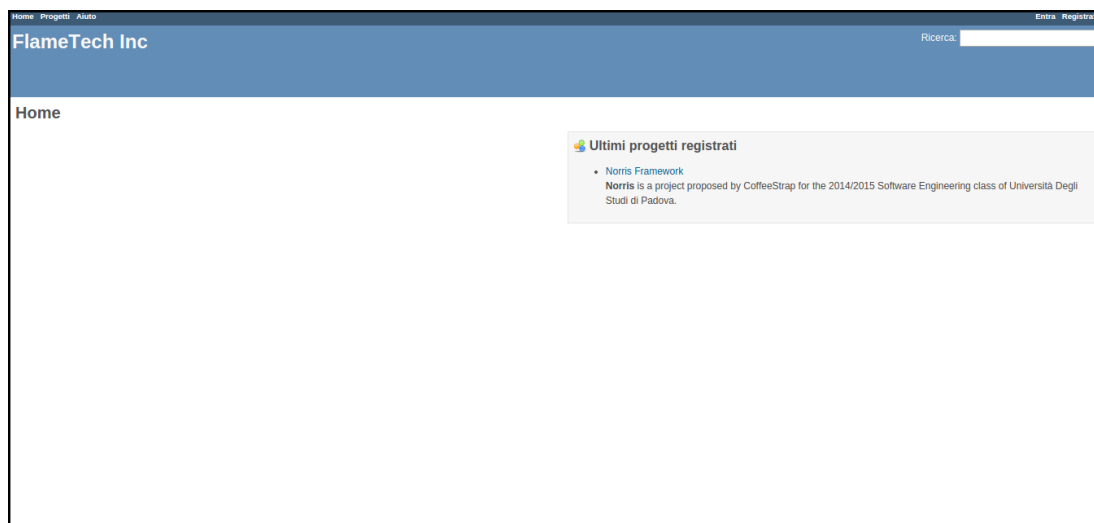


Figura 10: Server *Redmine<sub>G</sub>*

### 3.8.1.1 Creazione di un *ticket<sub>G</sub>*

Per la creazione di un *ticket<sub>G</sub>* il *Responsabile* deve seguire il seguente iter:

1. autenticarsi al *server<sub>G</sub>* <http://188.166.60.146/redmine>;
2. selezionare la scheda “Nuova segnalazione” nel menù in alto;
3. premere il bottone “Privato” in alto a destra;
4. compilare completamente l'apposita form, possono essere lasciati vuoti gli ultimi 4 campi: Checklist, Screenshot, File e Osservatore;
5. assicurarsi di aver salvato il *ticket<sub>G</sub>* premendo il tasto “Crea”.

Il software provvederà automaticamente ad informare l'assegnatario del *ticket<sub>G</sub>*, inserire il *ticket<sub>G</sub>* nel calendario e una volta terminato a creare il *Gantt<sub>G</sub>*. In questo modo il lavoro che dovrà esser svolto dal *Responsabile* risulterà alleggerito, ottimizzando anche le comunicazione tra i membri.

### 3.8.2 Gestione delle *issue<sub>G</sub>*

Per quanto concerne le segnalazioni di *bug<sub>G</sub>*, e la gestione delle stesse come richiesto dal Proponente nel capitolato d'appalto, si farà uso del sistema di gestione delle *issue<sub>G</sub>* integrato in *GitHub<sub>G</sub>*.

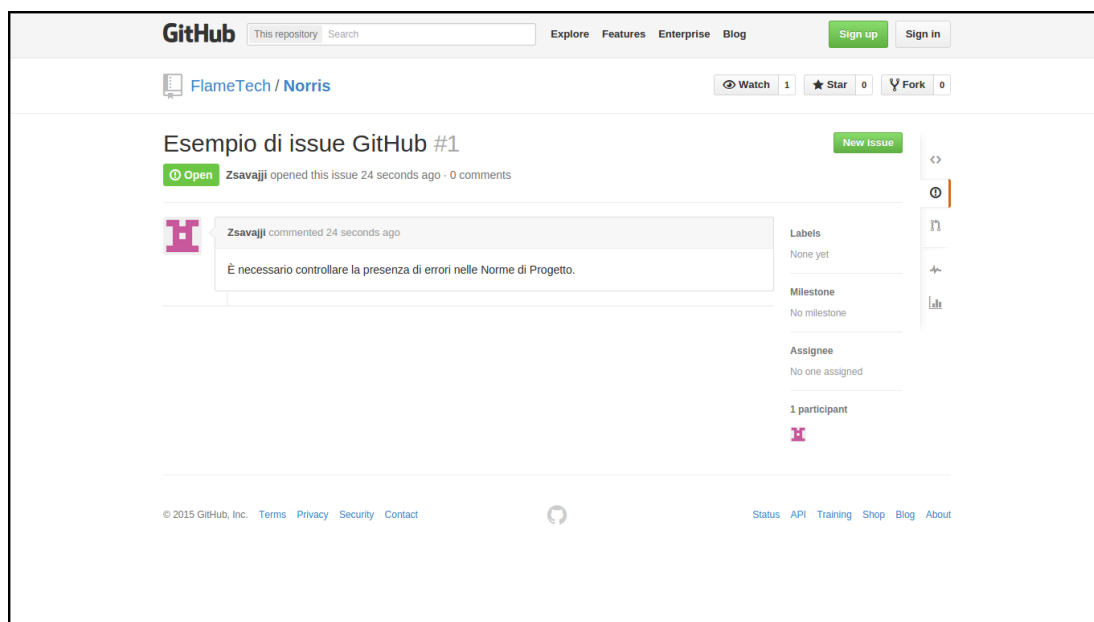


Figura 11: Esempio di *issue<sub>G</sub>*

### 3.8.2.1 Contenuto di una *issue<sub>G</sub>*

A una *issue<sub>G</sub>* possono essere associati:

- Immagini che descrivono in modo visuale il *bug<sub>G</sub>*;
- file con possibili soluzioni, sotto forma di modifiche o di nuovi file da aggiungere al *repository<sub>G</sub>*;
- *Milestone<sub>G</sub>* da rispettare per la soluzione del problema;
- Membri del gruppo a cui è stato assegnato il compito di risolvere il problema.

### 3.8.2.2 Creazione e completamento di *issue<sub>G</sub>*

Secondo il sistema fornito da *GitHub<sub>G</sub>*, una *issue<sub>G</sub>* può essere creata, vi si possono aggiungere commenti, può essere categorizzata tramite etichette, e può essere chiusa fornendo o meno una motivazione. Per aprire una *issue<sub>G</sub>* sarà necessario:

1. autenticarsi a *GitHub<sub>G</sub>* <http://github.com>;
2. entrare nel *repository<sub>G</sub>* del progetto;
3. selezionare la sezione *Issues<sub>G</sub>* dal menù di navigazione a destra;
4. selezionare l'opzione *New Issue<sub>G</sub>*, e procedere alla compilazione di titolo e descrizione (è possibile inserire del *markdown<sub>G</sub>*);
5. procedere all'inserimento mediante il pulsante *Submit new issue<sub>G</sub>*.

Per modificarla o chiuderla sarà invece necessario:

1. autenticarsi a *GitHub<sub>G</sub>* <http://github.com>;
2. entrare nel *repository<sub>G</sub>* del progetto;



3. selezionare la sezione *Issues<sub>G</sub>* dal menù di navigazione a destra;
4. selezionare l'*issue<sub>G</sub>* desiderata, e provvedere a modificare i campi necessari;
5. procedere alla modifica o alla chiusura.

I membri sono tenuti a fornire sempre una motivazione dettagliata per ogni cambiamento di stato delle *issue<sub>G</sub>*, sotto forma di commento. Al fine di facilitare la comprensione dell'ambito della modifica, nel titolo della *issue<sub>G</sub>* si deve inserire un' identificatore:

- **[BUG]**: per indicare che la *issue<sub>G</sub>* rappresenta un problema;
- **[MOD]**: per indicare una richiesta di modifica.

Sarà possibile non applicare l'identificatore, e in tal caso la *issue<sub>G</sub>* verrà considerata a bassa priorità. Il sistema provvederà a notificare tutti i membri iscritti al *repository<sub>G</sub>* tramite e-mail automaticamente ogni qualvolta la *issue<sub>G</sub>* subisca cambiamenti di stato.

## 4 Processi di supporto

### 4.1 Documentazione

In questo capitolo verranno descritte quali saranno le scelte tipografiche che il gruppo ha deciso di seguire nella realizzazione dei documenti.

#### 4.1.1 *Template<sub>G</sub>*

Per facilitare la verifica e uniformare la stesura dei documenti si è deciso di ricorrere alla realizzazione di un *template<sub>G</sub>* L<sup>A</sup>T<sub>E</sub>X<sub>G</sub>, il quale contiene tutte le impostazioni tipografiche che si è scelto di adottare. Questo si trova nel *repository<sub>G</sub>* vedi sez. 3.7.1. La scelta di creare un *template<sub>G</sub>* è stata fatta perché qualora si decidesse di apportare una modifica alla struttura dei documenti è sufficiente effettuarla su di un singolo *template<sub>G</sub>* affinché questa poi venga acquisita da tutti i file che ne fanno uso.

##### 4.1.1.1 Struttura del *Template<sub>G</sub>*

Il *template<sub>G</sub>* è stato realizzato con la seguente struttura:

**4.1.1.1.1 Prima Pagina** In prima pagina sono contenute le seguenti informazioni:

- Logo del gruppo;
- Titolo del documento;
- Nome del gruppo;
- Versione del documento;
- Lista dei redattori;
- Lista dei verificatori;
- Nome del Responsabile che ha approvato il documento;
- Destinazione del documento (uso interno/esterno);
- Lista di distribuzione;
- Descrizione del contenuto all'interno del documento.



Figura 12: Template di prima pagina

**4.1.1.1.2 Diario delle modifiche** Nella seconda pagina del documento è contenuta una tabella contenente il diario delle modifiche che vengono apportate al documento. Le informazioni riportate per ogni modifica sono:

- Stato del documento vedi sez. 4.1.4;
- Descrizione della modifica;
- Autore della modifica;
- Il ruolo della persona che ha modificato il documento;
- Data in cui è avvenuta la modifica;
- Versione del documento.

Le modifiche devono essere riportate in tabella secondo l'ordine decrescente di versione del documento. Inoltre deve essere aggiornata la versione del documento secondo le norme riportate in sez. 4.1.5. Questo comporta anche che sia aggiornata la versione del documento nel frontespizio qualora c'è ne sia la necessità.


 FLAMETECH Inc.					
Stato	Modifica	Autore	Ruolo	Data	Versione
Stato	Descrizione	Autore	Ruolo	AAAA/MM/GG	A.B.C
Stato	Descrizione	Autore	Ruolo	AAAA/MM/GG	A.B.C

Figura 13: Template del diario delle modifiche

**4.1.1.1.3 Indici** Nel *template<sub>G</sub>*, dopo il diario delle modifiche c'è la possibilità di inserire o meno i seguenti indici :

- Indice delle pagine;
- Indice delle tabelle;
- Indice delle figure.

L'inserimento dei seguenti indici è dato dalla presenza o meno di tabelle e/o figure all'interno del documento, qualora non fossero presenti tabelle o figure è sufficiente settare a false il flag di caricamento dell'indice corrispondente.

**4.1.1.1.4 Struttura delle pagine** Ogni pagina interna del documento avrà la seguente struttura:

- **Intestazione:** nella parte sinistra il logo del gruppo ed il nome mentre nella parte destra conterrà il numero e il nome della sezione;
- **Piè di pagina:** nella parte sinistra il nome del documento e la sua versione mentre nella parte destra è presente il numero della pagina. In numero di pagina è espresso nel formato: [numeroPagina] di [numeroPagineTotali];

## 4.1.2 Norme Tipografiche

Questa sezione contiene le scelte in fatto di tipografia, ortografia e stile che dovranno essere attuate durante la stesura dei documenti.

### 4.1.2.1 Stile del testo

- **Grassetto:** deve essere usato nei seguente casi:
  - **Elenchi:** per evidenziare la descrizione dell'elenco;
  - Header delle tabelle.
- **Corsivo:** solo nei seguenti casi:
  - **Documenti:** per citare un documento (es. *NormeDiProgetto\_ver3.0.0*);
  - **Ruoli:** per nominare uno dei ruoli del gruppo (es. *Analista*);
  - **Abbreviazioni:** quando c'è bisogno di inserire una abbreviazione di qualche termine;

- **Termini in glossario:** i termini presenti in glossario;
- **$URL_G$ :** per l'inserimento di indirizzi web verrà utilizzato l'apposito tag  $\text{\LaTeX}_G$ .

- **Maiuscolo:**

- L'inserimento di un termine tutto in maiuscolo è possibile solamente nel caso si tratti di acronimi;
- Per delle macro  $\text{\LaTeX}_G$  create appositamente per velocizzare la stesura dei documenti.

#### 4.1.2.2 Formati

- **Data:** le date dovranno adottare il formato  $aaaa/mm/gg$ , dove  $aaaa$  indica l'anno per esteso,  $mm$  il mese,  $gg$  il giorno;
- **Ruoli di progetto:** in corsivo, con la prima lettera maiuscola e poi tutto minuscolo;
- **Nomi propri:** i nomi dei componenti del team dovranno essere nella forma *Cognome Nome*;
- **Nome del gruppo:** per inserire il nome del gruppo usare l'apposita macro  $\text{\LaTeX}_G$ .

#### 4.1.2.3 Immagini

Tutte le immagini che vengono inserite all'interno del documento dovranno avere una didascalia significativa e singolare. Ogni immagine dovrà avere un numero progressivo che la identifichi così da poter essere più facilmente rintracciabile e dovrà essere inserita nell'indice delle immagini del documento. Tutte le immagini presenti nei file dovranno essere memorizzate in formato  $PNG_G$ ; è stato scelto questo formato poiché è compatibile con tutte le applicazioni usate dal gruppo, crea file di dimensioni ridotte ed è un formato *open source* con licenza  $GPL_G$ .

#### 4.1.2.4 Tabelle

Anche per le tabelle, al fine di renderle facilmente rintracciabili e comprensibili, dovranno essere identificate da un numero progressivo ed una didascalia singolare cosicché possano essere inserite nell'apposito indice. Per quanto riguarda la loro struttura grafica invece dovranno avere:

- **Header colonne:** testo nero in grassetto;
- **Celle generiche:** testo nero;
- **Colore di sfondo delle celle:** bianco.

#### 4.1.2.5 Elenchi

Gli elenchi potranno essere sia di tipo numerico che di tipo puntato. I numerici verranno usati quando l'ordine degli elementi è rilevante, mentre gli altri verranno usati quando c'è il bisogno di scrivere una semplice lista di elementi in cui l'ordine di apparizione è superfluo. Gli elenchi numerici potranno iniziare sia da 1, formattazione standard del  $\text{\LaTeX}$ , oppure da 0. La scelta sta al redattore del documento in base al contesto in cui si trova. Ogni punto di un elenco deve terminare con il punto e virgola, tranne l'ultimo che deve concludersi con il punto.

#### 4.1.2.6 Norme sui nomi dei file

Per rendere più agevole l'accesso ai file i membri sono tenuti ad utilizzare la seguente nomenclatura per tutti i file caricati nel *repository<sub>G</sub>*:

NomeDelFile\_verA.B.C.ext

Dove "NomeDelFile" indica il nome del file che deve iniziare con lettera maiuscola ed "ext" indica l'estensione del file.

Nel caso il nome sia composto da più parole verrà seguita la convenzione *CamelCase<sub>G</sub>*, la prima parola può iniziare sia con maiuscola che con la minuscola mentre la prima lettera di ogni parola successiva dovrà essere maiuscola. Per quanto riguarda la versione del file si rimanda alla sezione 4.1.5. Non vi è la possibilità di introdurre separatori nei nomi dei file o di utilizzare nomenclature che non rispettino queste regole. Inoltre non è permesso l'utilizzo di nomi contenenti lettere accentate.

Per uniformare e velocizzare la stesura dei file sono state create delle apposite macro per la citazione dei vari documenti:

- $\backslash\text{NdP}$ : Norme di Progetto;
- $\backslash\text{PdP}$ : Piano di Progetto;
- $\backslash\text{PdQ}$ : Piano di Qualifica;
- $\backslash\text{Glo}$ : Glossario;
- $\backslash\text{SdF}$ : Studio di Fattibilità;
- $\backslash\text{AR}$ : Analisi Requisiti;
- $\backslash\text{ST}$ : Specifica Tecnica.

#### 4.1.3 Strumenti per i documenti

Per le attività di documentazione sarà fatto uso di alcuni strumenti:

- $\text{\LaTeX}$ : i documenti saranno scritti utilizzando il linguaggio  $\text{\LaTeX}$ ;
- $\text{\TeXStudio}_G$ : per facilitare la scrittura del codice  $\text{\LaTeX}$ , sarà fatto uso di questo editor integrato;
- *Apache OpenOffice<sub>G</sub>*: sarà usato per la creazione dei grafici necessari nei documenti di pianificazione;

- **ProjectLibre<sub>G</sub>**: sarà usato per la creazione dei diagrammi di *Gantt<sub>G</sub>* usati nei documenti di pianificazione.

#### 4.1.4 Ciclo di vita del documento

Durante la realizzazione ogni documento si troverà in uno dei seguenti tre stati, anche più di una volta, fino ad arrivare alla sua versione finale. I tre stati in cui si potrà trovare sono:

- **In lavorazione:** un documento entra in questo stato nel momento della sua creazione e vi permane fino alla conclusione della sua realizzazione, oppure vi rientra nel caso siano necessarie successive modifiche.  
Una volta concluso, il documento rimarrà in questo stato fino al momento in cui i *Verificatori* non confermeranno che esso è privo di errori e non contiene delle imprecisioni, sia semantiche che sintattiche;
- **Verificato:** il documento entra in questo stato una volta accertato dai *Verificatori* che non sono presenti errori o incongruenze. Rimane in questo stato fino al momento in cui non riceve l'approvazione da parte del *Responsabile*.  
Se invece vengono riscontrati degli errori allora il documento ritornerà nello stato In lavorazione;
- **Approvato:** se la verifica ha dato esito positivo il documento dovrà essere approvato dal *Responsabile di Progetto*. L'approvazione ne decreta lo stato finale per la data versione.

Una volta approvato, se vi fosse la necessità di una revisione formale il documento rientrerà nello stato In lavorazione e dovrà affrontare nuovamente tutto l'iter che lo porterà ad una nuova approvazione e conseguentemente ad un avanzamento di versione. Lo stato di lavorazione del documento sarà annotata nel diario dello stesso.

#### 4.1.5 Versionamento

Ogni documento nel momento in cui sarà distribuito dovrà essere corredato dal numero di versione.

La codifica scelta per la versione è:

*verA.B.C*

con il significato:

- **A:** rappresenta il numero crescente di approvazioni del documento;
- **B:** rappresenta il numero crescente di modifiche sostanziali;
- **C:** rappresenta il numero crescente di modifiche minori effettuate.

La versione di ogni documento è visibile nel frontespizio e dovrà corrispondere con quella riportata nel nome del documento.

Per quanto riguarda la sua evoluzione invece, sarà possibile controllare tutti i passaggi nel "Diario delle modifiche" presente in ogni documento, che verrà aggiornato ad ogni modifica.

In particolare, l'avanzamento di versione avviene a seconda dello stato in cui si trova il documento in seguito alla modifica apportata e seguendo una modalità specifica:

- **In Lavorazione:** quando viene effettuata una modifica al contenuto del documento, l'avanzamento di versione viene applicato alla cifra  $C$  che viene incrementata di una unità;
- **Verificato:** quando viene effettuata la verifica del documento, l'avanzamento di versione viene applicato alla cifra  $B$  che viene incrementata di una unità, azzerando lo stato della cifra  $C$ ;
- **Approvato:** quando viene effettuata l'approvazione del documento, l'avanzamento di versione viene applicato alla cifra  $A$  che viene incrementata di una unità, azzerando lo stato delle cifre  $B$  e  $C$ .

#### 4.1.6 Classificazione dei documenti

Un documento realizzato sarà classificato come formale o informale, condizione dettata dallo stadio di avanzamento in cui si trova:

##### 4.1.6.1 Documento informale

Un documento informale è da considerarsi esclusivamente come un documento interno al gruppo, il documento sarà così classificato da quando viene creato fino al momento in cui riceve l'approvazione dal *Responsabile*. Questo tipo di documento sarà archiviato all'interno di un *repository* a cui hanno accesso solamente i membri del team.

##### 4.1.6.2 Documento formale

Una volta ricevuta l'approvazione dal *Responsabile*, il documento potrà essere classificato come formale e conseguentemente distribuito all'esterno del gruppo a chi ne facesse richiesta. Il passaggio da informale a formale coinciderà con le revisioni presso il Committente. Ogni documento per poter essere considerato come formale dovrà aver seguito tutto l'iter descritto in questo documento nella sezione 4.1.4.

#### 4.1.7 Glossario

Per non creare ambiguità a chi leggerà i documenti è stato redatto un glossario nel quale andranno inserite tutte le definizioni che potrebbero essere fraintese dal lettore.

##### 4.1.7.1 Inserimento del termine

L'inserimento del termine andrà fatto parallelamente alla stesura del documento. Qualora non fosse possibile per qualche motivo inserire immediatamente la definizione, il redattore dovrà inserire il termine lasciando la descrizione vuota, così facendo si ridurranno i possibili errori umani, ad esempio la possibilità che un termine segnato, in qualche documento, come presente a glossario sia poi tralasciato.

##### 4.1.7.2 Ordine di apparizione

I termini contenuti nel glossario seguiranno l'ordine alfabetico, sta quindi a chi andrà ad inserirlo accertarsi di rispettare l'ordine.



## 4.2 Verifica

La fase di verifica deve essere attuata fin dal principio e non relegata solamente alla fine, così facendo è possibile ottenere un risultato migliore. In questa sezione verranno descritte le metodologie che i *Verificatori* dovranno seguire per effettuare le analisi statiche e dinamiche dei processi, dei documenti e del codice.

### 4.2.1 Tecniche di Analisi

#### 4.2.1.1 Analisi Statica

Questa tecnica di analisi, che non richiede l'esecuzione del software, può essere applicata sia al codice che alla documentazione. La verifica effettuata con questa tecnica porta a rilevare eventuali anomalie ed errori prodotti. Per effettuarla si ricorre a due differenti metodologie che usualmente vengono applicate in maniera complementare.

**4.2.1.1.1 Inspection** L'attuazione di questa procedura implica un'analisi mirata solo di alcune sezioni del documento o del codice, cioè quelle che si ritiene probabili fonti del maggior numero di errori; risulta essere quindi molto più rapida del *Walkthrough*.

Essendo eseguita in modo mirato, come prima cosa deve avvenire una pianificazione di quali parti andare a sottoporre alla verifica, poi deve essere definita una lista di controllo e solo successivamente si può passare all'attività di verifica vera e propria.

Per questa procedura è richiesta una buona maturità da parte dei verificatori, acquisita nel tempo attraverso la tecnica del *Walkthrough*. Un prerequisito dell'*Inspection* è che le persone che andranno a verificare il documento non siano le stesse che precedentemente lo avevano redatto, ed è necessario che venga presa nota delle anomalie riscontrate o delle correzioni apportate, secondo quanto definito nel documento *NormeDiProgetto\_ver3.0.0*.

**4.2.1.1.2 Walkthrough** Con questo metodo si va ad eseguire una lettura a largo spettro, che di conseguenza impiega più tempo dell'*Inspection*. Questo tipo di controllo viene fatto principalmente nelle prime fasi del progetto, quando l'esperienza dei membri del gruppo non è sufficientemente elevata e quindi si rende necessario effettuare una verifica di più ampia portata. Da questa analisi i *Verificatori* sono in grado di capire quali sono gli errori più frequenti e conseguentemente di apportare dei miglioramenti nelle attività di verifica per le fasi successive.

Essendo necessario un controllo completo dell'intero documento è necessario un impiego di risorse maggiore. Terminata la fase di ricerca dei possibili errori verrà effettuata una discussione per esporre i difetti trovati ed organizzare la correzione. Solo al termine di quest'ultima si procederà all'applicazione effettiva dei provvedimenti decisi. Anche in questo caso è necessario che venga presa nota delle anomalie riscontrate o delle correzioni apportate, secondo quanto definito nel documento *NormeDiProgetto\_ver3.0.0*.

#### 4.2.1.2 Analisi dinamica

A differenza dell'analisi statica, l'analisi dinamica viene svolta solamente sul prodotto software e necessita della sua esecuzione. L'analisi verrà effettuata mediante lo svolgimento sul codice di test appositamente predisposti per individuare possibili malfunzionamenti o errori in fase di progettazione. Affinché l'analisi sia efficace, risulta

necessario che i test siano ripetibili; questo è fondamentale, poiché solo un test che, dato lo stesso input, produce il medesimo output è utile per verificare la corretta implementazione del codice. Per creare dei test ripetibili è quindi necessario definire a priori:

- **Ordine e procedure di esecuzione:** l'ordine e le modalità con cui i test devono essere eseguiti, e i metodi necessari all'analisi del risultato;
- **Relazione Input/Output:** gli input significativi per l'esecuzione e gli output attesi;
- **Ambiente di esecuzione:** il sistema hardware e software sul quale saranno effettuati questi test.

Sono stati individuati quindi cinque tipi di test: *test di unità*, *test di integrazione*, *test di regressione*, *test di sistema*, *test di accettazione*.

**4.2.1.2.1 Test di unità** Tramite i test di unità verrà controllata ogni singola componente creata per il prodotto. Questi test prevedono l'utilizzo di appositi *driver<sub>G</sub>*, *stub<sub>G</sub>* e *logger<sub>G</sub>*.

Lo scopo è quello di verificare il corretto funzionamento di ogni singola unità che compone il sistema e di eliminare i possibili errori commessi nel corso della programmazione. Durante questi test vengono rilevati circa i  $\frac{2}{3}$  dei difetti presenti. Con unità si intende la più piccola parte di software che è conveniente verificare autonomamente, tipicamente realizzata dallo stesso programmatore. L'individuazione degli errori in questa fase riduce la propagazione degli stessi nei test successivi e comporta una riduzione dei costi, poiché prima viene trovato l'errore e minore è il costo da sostenere per risolverlo.

**4.2.1.2.2 Test di integrazione** Tramite i test di integrazione verrà verificata la combinazione di due o più unità al fine di permettere corrette aggiunte incrementali al prodotto software. Con questo approccio i possibili malfunzionamenti saranno facilmente tracciabili, in quanto rende possibile dimostrare la completa funzionalità fino all'incremento precedente. In questi test potranno emergere errori dovuti a comportamenti non previsti nelle componenti software preesistenti e/o realizzate da terzi. Anche durante questi test sarà necessario ricorrere a parti fittizie di utilità, per sostituire le componenti non ancora realizzate.

**4.2.1.2.3 Test di regressione** Tramite i test di regressione verrà verificato, se necessario, che eventuali modifiche apportate a una o più componenti non pregiudichino il funzionamento delle componenti già verificate e dipendenti da queste ultime, che non sono state modificate. Per capire quali test di unità e/o integrazione debbano essere ripetuti a causa della modifica apportata è necessario riferirsi al tracciamento.

**4.2.1.2.4 Test di sistema** Tramite i test di sistema verrà verificato che il comportamento dinamico del sistema sia conforme ai requisiti software definiti nel documento *AnalisiRequisiti\_ver4.0.0*, e completo rispetto ad essi.

**4.2.1.2.5 Test di accettazione** Il test di accettazione verrà svolto in presenza del Proponente, e sarà il collaudo ufficiale del prodotto software finito. L'esito positivo dello stesso comporterà il rilascio ufficiale del prodotto.

## 4.2.2 Procedure

### 4.2.2.1 Metriche errori riscontrati e gestione cambiamenti

In questa sezione verranno definite delle metriche per gli errori che i *Verificatori* possono trovare, verranno forniti criteri per la quantificazione dell'impatto che questi potrebbero avere sul processo o sul prodotto e la definizione delle priorità per l'intervento. Così facendo sarà possibile stilare una lista e agire prima nella risoluzione degli errori che hanno impatto maggiore.

La gravità dell'errore può essere:

- **Bassa** se l'errore ha impatto su aspetti marginali del prodotto o provoca un basso aumento dei costi o dei tempi del processo;
- **Media** se l'errore ha impatto significativo sul prodotto o provoca un aumento percepibile di tempi e costi;
- **Alta** se l'errore rende il prodotto inutilizzabile o provoca un forte aumento dei tempi o dei costi.

Ambito	Gravità Bassa	Gravità media	Gravità alta
Errore nel prodotto	Impatto su aspetti marginali	Impatto su aspetti visibili	Prodotto inutilizzabile
Errore nei processi	Aumento costi o tempi <10%	Aumento costi o tempi <25%	Aumento costi o tempi >25%

Tabella 2: Gravità dell'errore e impatto su processi e prodotti

La priorità di risoluzione può essere:

- **Entro la *milestone<sub>G</sub>***: indica che l'errore deve essere risolto prima della *milestone<sub>G</sub>* successiva;
- **Breve**: indica che l'errore deve essere risolto entro 4-5 giorni
- **Urgente**: indica che l'errore deve essere risolto appena possibile.

Le modalità operative per il *Verificatore* sono le seguenti:

- ***Ticket<sub>G</sub>***: il *Verificatore* deve aprire un *ticket<sub>G</sub>* segnalando il problema al *Responsabile* che provvederà a designare le modalità e le persone addette alla correzione;
- **Correzione immediata**: è richiesto che il *Verificatore* proceda autonomamente alla correzione dell'errore;
- **Aggiunta alla checklist**: è richiesto che il *Verificatore* aggiunga l'errore riscontrato alla checklist appropriata.

Errore	Gravità	Modalità Operativa	Priorità
Errore tracciamento di <i>casi d'uso<sub>G</sub></i> e requisiti	Alta	<i>Ticket<sub>G</sub></i>	Urgente
Errore di progettazione	Alta	<i>Ticket<sub>G</sub></i>	Urgente
Indici fuori range	Alta	<i>Ticket<sub>G</sub></i>	Urgente
Ritardi superiori ai 3 giorni	Alta	<i>Ticket<sub>G</sub></i>	Urgente

Tabella 3: Errori nei processi e modalità di risoluzione

Errore	Gravità	Modalità Operativa	Priorità
Errore ortografico	Bassa	<i>Ticket<sub>G</sub></i>	Entro la <i>Milestone<sub>G</sub></i>
Compilazione del documento fallita	Media	<i>Ticket<sub>G</sub></i>	Urgente
Valore <i>Gulpease<sub>G</sub></i> fuori range	Alta	<i>Ticket<sub>G</sub></i>	Urgente
Errore formalismo UML 2.0	Alta	<i>Ticket<sub>G</sub></i>	Urgente
Errore sistematico di ortografia	Media	<i>Ticket<sub>G</sub></i>	Urgente
Mancata compilazione del codice	Alta	Correzione immediata	Urgente

Tabella 4: Errori nei documenti e modalità di risoluzione

#### 4.2.2.2 Verifica dei processi

Il protocollo per l'analisi dei processi dovrà eseguire tutti i passi di seguito descritti nell'ordine di apparizione:

##### 1. Controllo valore indici

Terminata ogni macro attività, come definita nel *Piano di Progetto*, dovranno essere verificati i valori degli indici *BV<sub>G</sub>* ed *SV<sub>G</sub>*. Per una buona misurazione sarà necessario valutarne la media;

##### 2. Analisi grafico *PDCA<sub>G</sub>*

Si dovrà anche andare a valutare il grafico del *PDCA<sub>G</sub>* e da qui si potranno trarre delle conclusioni sullo svolgimento dei processi. Dall'analisi di questo grafico il *Responsabile* potrà capire se sono stati fatti errori durante la fase di pianificazione del lavoro e la velocità con cui procede il gruppo nel portare avanti i processi.

#### 4.2.2.3 Verifica dei documenti

Per un controllo efficiente ed accurato dei documenti dovranno essere eseguiti tutti i passi nel seguente ordine:

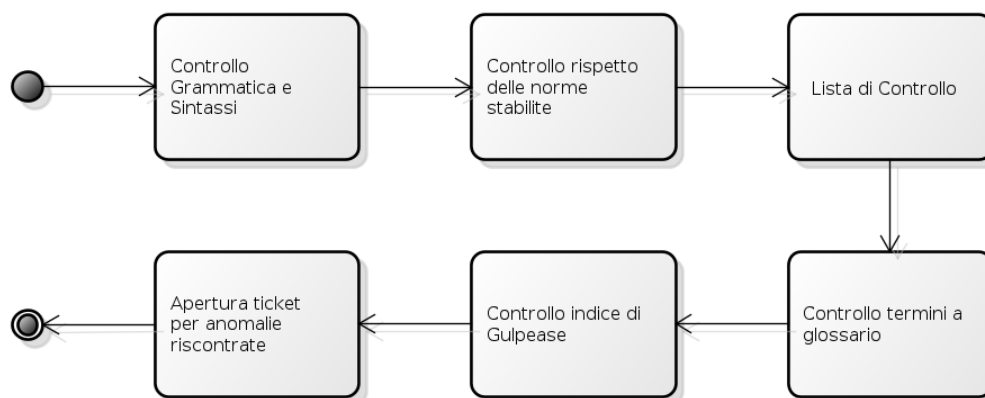


Figura 14: Procedura per l'analisi dei documenti

### 1. Controllo grammatica e sintassi

Questo controllo viene eseguito in parte in maniera automatica, con l'ausilio di software, ed in parte deve essere svolto manualmente dal verificatore. Per la correzione degli errori grammaticali e di sintassi è stato usato il correttore di *TeXstudio<sub>G</sub>*, dopo aver importato il dizionario italiano. Per quanto riguarda invece la punteggiatura e la comprensione delle frasi complesse deve intervenire manualmente un *Verificatore*;

### 2. Norme di Progetto

La verifica delle norme dovrà essere svolta manualmente da parte del *Verificatore*. Egli dovrà inoltre controllare che quanto scritto nei vari documenti corrisponda a ciò che è stato stabilito nel documento *NormeDiProgetto\_ver3.0.0*;

### 3. Lista controllo

Il *Verificatore* dovrà utilizzare la lista di controllo per i documenti al fine di verificare che gli errori tipici non siano presenti;

### 4. Termini a glossario

Il controllo della corrispondenza tra i termini segnalati come presenti a glossario e la presenza effettiva, almeno nella fase iniziale, dovrà essere svolta manualmente, e verrà successivamente demandata a uno script automatico che verifichi eventuali mancanze o corrispondenze non segnalate;

### 5. Indice di *Gulpease<sub>G</sub>*

Per ogni documento dovrà essere verificato che l'indice di leggibilità del documento rientri nel range di accettazione, in caso negativo si dovrà provvedere alla modifica ed ad una successiva ulteriore verifica;

### 6. Segnalazione anomalie

Il *Verificatore* terminata l'analisi del documento dovrà segnalare le anomalie riscontrate attraverso l'attivazione dei *ticket<sub>G</sub>*, come descritto nella sezione *ticket<sub>G</sub>*.

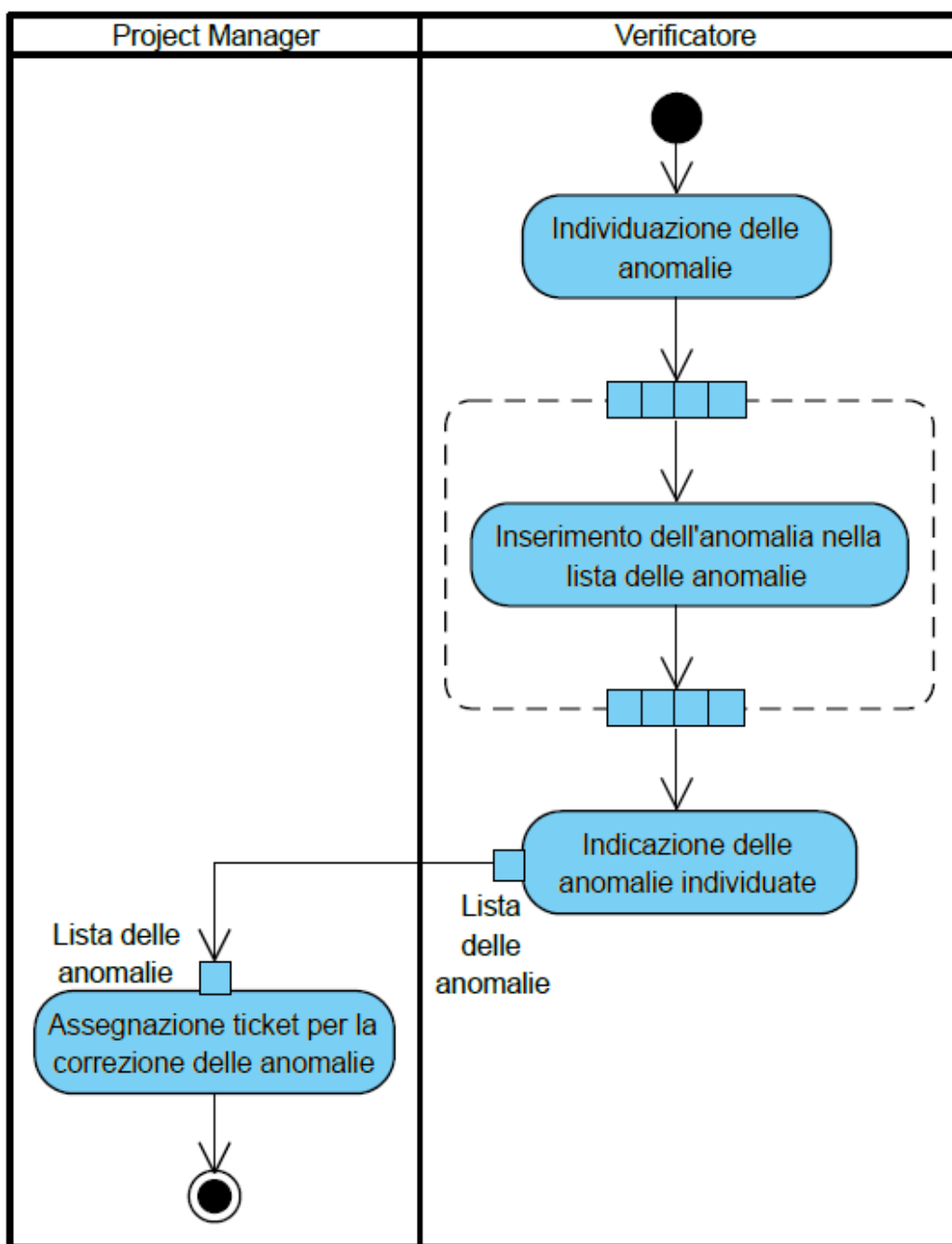


Figura 15: Gestione delle anomalie

#### 4.2.2.4 Verifica tracciamento dei requisiti

Il *Verificatore* deve accertarsi della corretta realizzazione dei *casi d'uso<sub>G</sub>* prodotti ed il corretto tracciamento con i requisiti, facendo riferimento ai dati contenuti in *FIRE<sub>G</sub>*.

#### 4.2.2.5 Verifica *UML<sub>G</sub>*

Il verificatore deve accertarsi che i diagrammi prodotti siano conformi alle specifiche *UML<sub>G</sub> 2.0*.

- **Diagrammi dei casi d'uso<sub>G</sub>:** questo tipo di controllo deve essere effettuato in maniera manuale dal *Verificatore*. Un punto importante su cui prestare attenzione è l'uso delle relazioni di inclusione e estensione. Inoltre deve accertarsi che quanto descritto nel caso d'uso sia effettivamente rappresentato nel diagramma.
- **Diagrammi delle classi:** al *Verificatore* è richiesto di controllare che i diagrammi rispettino il formalismo *UML<sub>G</sub>* 2.0, controllare la corrispondenza tra questi e la progettazione. È necessario inoltre controllare che essi siano ben formati e siano stati realizzati rispettando lo standard.

#### 4.2.2.6 Verifica progettazione architetturale

Il verificatore dovrà controllare che le metriche di accoppiamento efferente e afferente rispettino i range stabiliti.

#### 4.2.2.7 Verifica progettazione di dettaglio

Il documento *Definizione di Prodotto* dovrà essere controllato dal *Verificatore*, che dovrà accertarsi che questo contenga l'architettura di dettaglio con i vari moduli. Per ogni modulo dovrà accertarsi che abbia una dimensione e un accoppiamento che lo renda di facile implementazione per il *Programmatore*. Ogni modulo realizzato dovrà far riferimento ad almeno un requisito e si dovrà verificare la correttezza del tracciamento prodotto da *FIRE<sub>G</sub>*.

#### 4.2.2.8 Verifica del codice

Per il controllo del codice il *Verificatore* dovrà avvalersi dell'aiuto di test dinamici e statici, inoltre dovrà effettuare una attenta analisi dei risultati ottenuti.

### 4.2.3 Strumenti

#### 4.2.3.1 Strumenti per la verifica dei documenti

Per il processo di verifica dei documenti verranno usati i seguenti strumenti software:

- **Correttore *TeXstudio<sub>G</sub>*:** l'ambiente per la stesura dei documenti in *LaTeX<sub>G</sub>*, permette l'integrazione dei dizionari di *Apache OpenOffice<sub>G</sub>* che segnalano possibili errori ortografici;
- ***Aspell<sub>G</sub>*:** è un software libero per il controllo ortografico.

#### 4.2.3.2 Strumenti per la verifica del codice

Per il processo di verifica del codice verranno usati i seguenti strumenti software:

- ***JSHint<sub>G</sub>*:** è uno strumento per il rilevamento di errori e potenziali problemi nel codice *JavaScript<sub>G</sub>*, e per imporre convenzioni di codifica;
- ***Jasmine<sub>G</sub>*:** è un *framework<sub>G</sub>* di testing *open source<sub>G</sub>* per *JavaScript<sub>G</sub>*. Non dipende da altri *framework<sub>G</sub>* *JavaScript<sub>G</sub>* e non richiede un *DOM<sub>G</sub>*;
- ***Jenkins<sub>G</sub>*:** per automatizzare e velocizzare la verifica del codice ad ogni *push<sub>G</sub>* nel *repository<sub>G</sub>* verranno lanciati dei controlli automatici, che solamente in caso di esito positivo permetteranno di effettuare il *commit<sub>G</sub>* del nuovo codice nel

*repository<sub>G</sub>*. Grazie a questo strumento il *Verificatore* dovrà solamente consultare la *dashboard<sub>G</sub>* fornita dallo stesso;

- ***JSCover<sub>G</sub>***: è uno strumento utile per il calcolo della copertura del codice. Attraverso un eseguibile *Java<sub>G</sub>* sarà possibile analizzare i file sorgente e lo strumento evidenzierà quali sono gli *statement<sub>G</sub>* che non vengono eseguiti durante i test;
- ***JSMeter<sub>G</sub>***: è uno strumento per l'analisi del codice *JavaScript<sub>G</sub>*, che permetterà di capire se alcune delle metriche descritte nel *PianoDiQualifica\_ver3.0.0* siano state rispettate o meno. Lo strumento ci fornirà indicazioni su:
  - linee di codice;
  - linee di commento;
  - numero di *statement<sub>G</sub>*;
  - rapporto tra linee di codice e linee di commento;
  - indice di complessità ciclomatica.

Il software fornisce anche degli altri indici che il gruppo **FlameTech Inc.** non ritiene essere importanti.



## Appendici

### A Lista Controllo

Durante l'applicazione del walkthrough ai documenti, sono state riportate le tipologie di errori più frequentemente rilevate. 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 negli elementi interni;
  - Elenco puntato: l'ultimo elemento non termina con il punto;
  - Nome proprio di persona: non rispetta la forma Cognome Nome;
  - Nome documento: non viene utilizzata la macro predisposta;
  - Elenco numerato: non termina con il punto e virgola negli elementi interni;
  - Elenco numerato: l'ultimo elemento non termina con il punto;
  - Nome ruolo di progetto: non è in corsivo e/o non ha l'iniziale maiuscola;
  - Parole Proponente e Committente: non hanno l'iniziale maiuscola.
- 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 corretta;
  - Consistenza tra la nomenclatura dei diagrammi e le descrizioni testuali nei documenti.
- Ortografia:
  - Virgola tra soggetto e verbo: rende di difficile comprensione la frase;
  - Periodi: frasi troppo lunghe rendono i concetti di difficile comprensione;
  - Doppie negazioni: evitare l'utilizzo di doppie negazioni perché complicano la comprensione della frase;
  - Punto e virgola: evitare l'uso del punto e virgola quando è necessario usare il punto;
  - Proponente e Committente: non si deve confondere il loro significato.
- $\LaTeX$ :
  - Carattere di spaziatura: non deve essere utilizzato all'interno dei tag;
  - Macro  $\LaTeX$ : deve essere scritta usando l'apposito comando  $\LaTeX$ .
- Codice Javascript
  - mancato inserimento del carattere ";" nella terminazione degli statement;
  - utilizzo non corretto degli operatori "=", "==", "===", "!", "!==";
  - invocazione di metodi con parametri di tipo non corrisponde a quello atteso.

La seguente lista di controllo vuole riassumere invece gli errori più frequenti rilevati durante il walkthrough del tracciamento requisiti effettuato mediante il software *FIRE<sub>G</sub>*:

- Controllare in *FIRE<sub>G</sub>* che le fonti dei requisiti siano corrette;
- I codici dei *casì d'uso<sub>G</sub>* nei diagrammi e in *FIRE<sub>G</sub>* devono corrispondere;
- La fonte *Capitolato* non deve comparire nei requisiti interni;
- I requisiti devono essere una copertura totale del capitolato;
- Devono essere impostate le corrette relazioni di parentela tra requisiti in *FIRE<sub>G</sub>*;
- Ad ogni caso d'uso deve corrispondere almeno un requisito;
- Ad ogni requisito deve corrispondere almeno una fonte;
- Devono essere impostate le corrette relazioni di parentela tra *casì d'uso<sub>G</sub>* in *FIRE<sub>G</sub>*;
- In *FIRE<sub>G</sub>* deve essere indicato almeno un attore in ogni *caso d'uso<sub>G</sub>*;