



# *Norme di progetto v1.0.0*

*WarMachine – Progetto IronWorks*

warmachine.swe@gmail.com

## Informazioni sul documento:

<b>Versione</b>	1.0.0
<b>Data di creazione</b>	03/03/2018
<b>Redazione</b>	Cisternino Nicola, Zanetti Ilenia
<b>Verifica</b>	Zanon Elena, Coletti Andrea
<b>Approvazione</b>	Bragagnolo Leonardo
<b>Uso</b>	Interno
<b>Distribuzione</b>	<i>WarMachine</i>



## Diario delle modifiche

Versione	Data	Collaboratori	Ruolo	Descrizione
1.0.0	09/03/2018	Bragagnolo Leonardo	<i>Responsabile di progetto</i>	Approvazione del documento.
0.3.0	09/03/2018	Coletti Andrea, Zanon Elena	<i>Verificatore</i>	Verificato intero documento.
0.2.2	09/03/2018	Cisternino Nicola	<i>Amministratore</i>	Aggiunta metodo di miglioramento continuo dei processi: ciclo di deming <sub>G</sub> . Aggiunta descrizione Astah e PragmaDB.
0.2.1	09/03/2018	Cisternino Nicola	<i>Amministratore</i>	Aggiunti test unità, sistema <sub>G</sub> , integrazione, regressione e accettazione. Modifiche SPI e CPI con SV e CV.
0.1.1	08/03/2018	Cisternino Nicola	<i>Amministratore</i>	Correzioni grammaticali e di sintassi alle sezioni [1], [2], [3], [4]. Sezione 4.1.9.5 rivista e corretta.
0.2.0	08/03/2018	Coletti Andrea	<i>Verificatore</i>	Rivedere grammatica e sintassi, in particolare le sezioni: [1], [2], [3] e [4].
0.0.8	07/03/2018	Zanetti Ilenia	<i>Amministratore</i>	Correzione sezione [4].



0.1.2	07/03/2018	Cisternino Nicola	<i>Amministratore</i>	Aggiunte sigle dei nominativi dell'analisi dei rischi <sub>G</sub> sul <i>Piano di progetto</i> .
0.1.1	06/03/2018	Zanon Elena	<i>Verificatore</i>	Verificate sezioni: [1], [2], [3]. Da rivedere sezione [4], in particolare [4.1.5.1] Comunicazioni interne.
0.1.0	06/03/2018	Cisternino Nicola	<i>Amministratore</i>	Aggiunta metrica per errori concettuali nel testo [3.2]. Correzioni errori non conforme alle norme stabilite e errori di ortografia.
0.0.5	05/03/2018	Zanetti Ilenia	<i>Amministratore</i>	Modificati canali di coordinamento, aggiunta sottosezione [2.2.4] Analisi dei requisiti.
0.0.4	05/03/2018	Cisternino Nicola	<i>Amministratore</i>	Correzione incomprensioni processi di supporto.
0.0.3	04/03/2018	Cisternino Nicola, Zanetti Ilenia	<i>Amministratore</i>	Completata sezione [3], sezione [4] aggiunta di [4.1.9.7], revisione attività <sub>G</sub> di analisi in [3.2].
0.0.2	04/03/2018	Cisternino Nicola	<i>Amministratore</i>	Scritte sezioni [3] e [4].



---

0.0.1	03/03/2018	Cisternino Nicola	<i>Amministratore</i>	Creazione documento. Scritte sezioni [1] e [2].
-------	------------	-------------------	-----------------------	---



# Indice

<b>1</b>	<b>Introduzione</b>	<b>9</b>
1.1	Scopo del Documento . . . . .	9
1.2	Struttura del documento . . . . .	9
1.3	Scopo del prodotto <sub>G</sub> . . . . .	9
1.4	Glossario . . . . .	9
1.5	Riferimenti . . . . .	10
1.5.1	Normativi . . . . .	10
1.5.2	Informativi . . . . .	10
<b>2</b>	<b>Processi primari</b>	<b>11</b>
2.1	Processo <sub>G</sub> di fornitura . . . . .	11
2.1.1	Scopo del processo . . . . .	11
2.1.2	Descrizione contenuto . . . . .	11
2.1.3	Accordo contrattuale . . . . .	11
2.1.4	Modalità di esecuzione del contratto . . . . .	11
2.1.5	Consegna e manutenzione del prodotto . . . . .	11
2.1.6	Studio di Fattibilità . . . . .	12
2.2	Processo di sviluppo . . . . .	12
2.2.1	Scopo del processo . . . . .	12
2.2.2	Aspettative . . . . .	12
2.2.3	Descrizione contenuto . . . . .	12
2.2.4	Implementazione del processo . . . . .	13
2.2.5	Analisi dei requisiti . . . . .	13
2.2.5.1	Scopo dell'attività . . . . .	13
2.2.5.2	Aspettative . . . . .	13
2.2.5.3	Descrizione dell'attività . . . . .	13
2.2.5.4	Strumenti utili al tracciamento . . . . .	13
2.2.5.5	Casi d'uso . . . . .	13
2.2.5.6	Codice identificativo . . . . .	14
2.2.5.7	Requisiti . . . . .	14
2.2.5.8	Codice identificativo . . . . .	14
<b>3</b>	<b>Processi di supporto</b>	<b>16</b>
3.1	Processo di documentazione . . . . .	16
3.1.1	Scopo del processo . . . . .	16
3.1.2	Aspettative . . . . .	16
3.1.3	Descrizione contenuto . . . . .	16
3.1.4	Procedura di approvazione dei documenti . . . . .	16
3.1.5	Template L <sup>A</sup> T <sub>E</sub> X . . . . .	16
3.1.6	Struttura dei documenti . . . . .	17
3.1.6.1	Costruzione di un documento . . . . .	17
3.1.6.2	Prima pagina . . . . .	17
3.1.6.3	Struttura comune delle pagine . . . . .	17
3.1.6.4	Indice . . . . .	17
3.1.6.5	Diario delle modifiche . . . . .	18



3.1.7	Norme tipografiche . . . . .	18
3.1.7.1	Elenchi puntati . . . . .	18
3.1.7.2	Comandi personalizzati L <sup>A</sup> T <sub>E</sub> X . . . . .	18
3.1.7.3	Sigle e abbreviazioni . . . . .	20
3.1.7.4	Stile del testo . . . . .	21
3.1.7.5	Formati . . . . .	21
3.1.8	Componenti grafiche . . . . .	22
3.1.8.1	Tabelle . . . . .	22
3.1.8.2	Immagini . . . . .	22
3.1.9	Classificazione dei documenti . . . . .	22
3.1.9.1	Documenti informali . . . . .	22
3.1.9.2	Documenti formali . . . . .	22
3.1.9.3	Verbali . . . . .	22
3.1.10	Strumenti . . . . .	23
3.1.10.1	L <sup>A</sup> T <sub>E</sub> X . . . . .	23
3.1.10.2	TexMaker . . . . .	23
3.1.10.3	Astah . . . . .	23
3.1.10.4	Lucidchart . . . . .	24
3.1.10.5	Microsoft Excel . . . . .	24
3.2	Processo di gestione delle versioni e delle configurazioni . . . . .	24
3.2.1	Versionamento della documentazione . . . . .	24
3.2.2	Gestione del versionamento . . . . .	25
3.2.2.1	Repository . . . . .	25
3.2.2.2	Struttura del repository . . . . .	25
3.2.2.3	Tipi di file e .gitignore . . . . .	25
3.2.2.4	Norme sui commit . . . . .	26
3.3	Processo di verifica . . . . .	26
3.3.1	Scopo del processo . . . . .	26
3.3.2	Aspettative . . . . .	26
3.3.3	Descrizione del processo . . . . .	26
3.3.4	Linee guida per la verifica della documentazione . . . . .	26
3.3.5	Metodi di segnalazione degli errori . . . . .	27
3.3.5.1	Template GitHub Issues per segnalazione degli errori . . . . .	27
3.3.6	Analisi . . . . .	28
3.3.6.1	Analisi statica . . . . .	28
3.3.6.2	Analisi dinamica . . . . .	28
3.3.7	Test di verifica . . . . .	28
3.3.7.1	Test di unità[TU] . . . . .	28
3.3.7.2	Test di integrazione[TI] . . . . .	29
3.3.7.3	Test di sistema[TS] . . . . .	29
3.3.7.4	Test di regressione[TR] . . . . .	29
3.3.7.5	Test di accettazione[TA] . . . . .	29
3.3.8	Strumenti . . . . .	29
3.3.8.1	Verifica ortografica . . . . .	29
3.3.8.2	GitHub Issues . . . . .	29
3.3.8.3	Travis CI . . . . .	30



3.3.9	Metriche . . . . .	31
3.3.10	Metriche per i processi . . . . .	31
3.3.10.1	Modello Spice <sub>G</sub> . . . . .	31
3.3.10.2	Ciclo Di Deming . . . . .	33
3.3.10.3	SV - Schedule Variance <sub>G</sub> . . . . .	33
3.3.10.4	CV - Cost Variance <sub>G</sub> . . . . .	34
3.3.11	Metriche per i documenti . . . . .	34
3.3.11.1	Indice di Gulpease . . . . .	35
3.3.11.2	Indice di errori concettuali . . . . .	35
<b>4</b>	<b>Processi organizzativi</b>	<b>36</b>
4.1	Processo di gestione di progetto . . . . .	36
4.1.1	Scopo . . . . .	36
4.1.2	Aspettative . . . . .	36
4.1.3	Descrizione . . . . .	36
4.1.4	Ruoli di progetto . . . . .	36
4.1.4.1	Amministratore di progetto . . . . .	37
4.1.4.2	Responsabile di progetto . . . . .	37
4.1.4.3	Analista . . . . .	37
4.1.4.4	Progettista . . . . .	37
4.1.4.5	Verificatore . . . . .	38
4.1.4.6	Programmatore . . . . .	38
4.1.5	Gestione delle comunicazioni . . . . .	38
4.1.5.1	Comunicazioni interne . . . . .	38
4.1.5.2	Comunicazioni esterne . . . . .	39
4.1.6	Gestione degli incontri . . . . .	40
4.1.6.1	Incontri interni . . . . .	40
4.1.6.2	Incontri esterni . . . . .	41
4.1.7	Strumenti di coordinamento . . . . .	42
4.1.7.1	Ticketing . . . . .	42
4.1.8	Gestione dei rischi . . . . .	44
4.1.9	Strumenti . . . . .	44
4.1.9.1	Sistemi operativi . . . . .	44
4.1.9.2	Slack . . . . .	44
4.1.9.3	Telegram . . . . .	45
4.1.9.4	Asana . . . . .	45
4.1.9.5	GitHub . . . . .	46
4.1.9.6	GitHub Desktop . . . . .	46
4.1.9.7	GanttProject . . . . .	47
4.2	Processo di formazione . . . . .	47

## Elenco delle figure

1	Immagine esempio TexMaker. . . . .	23
2	Immagine esempio Lucidchart. . . . .	24
3	Immagine che illustra una delle funzionalità di Travis CI. . . . .	30



4	Immagine esplicativa per ISO/IEC 15504. . . . .	32
5	Procedura per l'organizzazione di un ritrovo interno. . . . .	40
6	Procedura per l'organizzazione di un ritrovo esterno. . . . .	41
7	Procedura per l'assegnazione di un ticket. . . . .	43
8	Immagine rappresentativa del profilo Slack del Team WarMachine. . . . .	45
9	Asana visualizzato sul web. . . . .	46
10	GitHub Desktop su Windows. . . . .	47





# 1 Introduzione

## 1.1 Scopo del Documento

Questo documento sancisce le norme di comportamento che verranno adottate dai membri del team<sub>G</sub> *WarMachine* nello svolgimento del progetto<sub>G</sub> *IronWorks*. All'interno del documento verranno trattati in particolare i seguenti punti:

- Regole d'interazione e comportamento tra membri del Team;
- Stesura e revisione di documenti;
- Ambiente di lavoro.

## 1.2 Struttura del documento

Il documento è strutturato seguendo i passi del modello standard Iso/IEC 12207. Il documento si compone di un'introduzione seguita da tre sezioni principali:

- Sezione adibita alla trattazione dei processi primari:
  - *Fornitura*;
  - *Sviluppo*.
- Sezione adibita alla trattazione dei processi di supporto:
  - *Gestione della documentazione*;
  - *Gestione della configurazione*;
  - *Verifica*;
  - *Validazione*.
- Sezione adibita alla trattazione dei processi organizzativi:
  - *Gestione di progetto*;
  - *Formazione*.

## 1.3 Scopo del prodotto<sub>G</sub>

Con lo sviluppo di questo progetto si intende creare un software capace di:

- Creare e modificare diagrammi di robustezza<sub>G</sub>;
- auto-generare codice Java<sub>G</sub> e Sql<sub>G</sub> a partire dal diagramma di robustezza realizzato precedentemente.

## 1.4 Glossario

Per evitare ogni sorta di ambiguità relativa al linguaggio utilizzato è stato creato un *Glossario* contenente le parole ambigue e di difficile comprensione. Queste parole sono marcate con una G nel pedice<sub>G</sub>.



## 1.5 Riferimenti

### 1.5.1 Normativi

- ISO/IEC 12207: [https://it.wikipedia.org/wiki/Iso\\_12207](https://it.wikipedia.org/wiki/Iso_12207);
- ISO/IEC 15504: [https://en.wikipedia.org/wiki/ISO/IEC\\_15504](https://en.wikipedia.org/wiki/ISO/IEC_15504);
- Capitolato<sub>G</sub> trattato: <http://www.math.unipd.it/~tullio/IS-1/2017/Progetto/C5.pdf>.

### 1.5.2 Informativi

- Guida fornita dal corso di  $\text{\LaTeX}$ 2016-2017 tenuto dal prof. Bresolin : <https://github.com/R-and-LaTeX/GuidaGalatticaPerLaTeX/releases/>;
- Software Engineering - Ian Sommerville - 9th Edition : in particolare il capitolo 4 - Requirements engineering;
- Slide del corso: <http://www.math.unipd.it/~tullio/IS-1/2017/>;
- Use-Case<sub>G</sub> 2.0 The Guide to Succeeding with Use Cases - Ivar Jacobson;
- Use Case Driven Object modeling - Doug Rosenberg, Matt Stephens.

## 2 Processi primari

### 2.1 Processo<sub>G</sub> di fornitura

#### 2.1.1 Scopo del processo

Lo scopo del processo di fornitura è quello di stabilire un accordo tra proponente<sub>G</sub> e fornitore, in cui quest'ultimo si impegna a consegnare un prodotto che soddisfa i requisiti concordati.

#### 2.1.2 Descrizione contenuto

Facendo riferimento allo standard Iso/IEC 12207:1995 il fornitore deve svolgere le seguenti attività:

- Accordo Contrattuale;
- Studio di fattibilità;
- Modalità di esecuzione del contratto.

#### 2.1.3 Accordo contrattuale

Il fornitore deve accordarsi con il proponente, Zucchetti s.p.a., per definire e contrattare le richieste presenti nel documento di presentazione del capitolato. La manutenzione del suddetto prodotto non rientra nei compiti del fornitore.

#### 2.1.4 Modalità di esecuzione del contratto

Il fornitore si impegna a collaborare con il proponente Zucchetti s.p.a. per l'intera durata del progetto per raggiungere i seguenti obiettivi:

- Concordare e chiarire i vincoli sui requisiti;
- Concordare e chiarire i vincoli di progetto.

Il fornitore si impegna a fornire al proponente, *Zucchetti s.p.a.*, e al committente<sub>G</sub>, *Prof. Vardanega Tullio e Prof. Cardin Riccardo*, la seguente documentazione:

- Piano di progetto;
- Piano di qualifica;
- Analisi dei requisiti.

#### 2.1.5 Consegna e manutenzione del prodotto

Il prodotto finito candidato all'accettazione dovrà essere consegnato su supporto CD-ROM / DVD, comprensivo di:

- Utilità d'installazione e istruzioni per l'uso;

- Sorgenti completi e utilità di compilazione;
- Documentazione completa e finale;
- Eventuali utilità di collaudo.

Il fornitore, *WarMachine*, come stabilito nell'accordo contrattuale non si occuperà della manutenzione del prodotto.

### 2.1.6 Studio di Fattibilità

È il primo documento da redigere per la revisione dei requisiti.

Il *Responsabile di progetto* deve organizzare delle riunioni con lo scopo di valutare i capitolati. Dalle riunioni deve emergere la stesura dello *Studio di fattibilità*, basato su:

- **Dominio tecnologico e applicativo:** Conoscenza delle tecnologie richieste dal capitolato;
- **Rapporto costi/benefici:** Prodotti simili già presenti sul mercato, quantità di requisiti obbligatori, costi in rapporto ai risultati stimati;
- **Individuazione dei rischi:** Comprensione dei punti critici della realizzazione quali mancanza di conoscenze o competenze e difficoltà nell'individuazione dei requisiti.

## 2.2 Processo di sviluppo

### 2.2.1 Scopo del processo

Il processo ha lo scopo di sviluppare un prodotto software che rispecchi la volontà del proponente. Di seguito verranno trattate le attività e i compiti svolti dal Team.

### 2.2.2 Aspettative

Le aspettative per la corretta esecuzione di questo processo sono:

- Realizzare un prodotto finale che sia in grado di soddisfare sia i test di validazione e verifica sia i requisiti concordati con il proponente;
- Fissare gli obiettivi per lo sviluppo;
- Fissare i vincoli tecnologici.

### 2.2.3 Descrizione contenuto

Il processo di sviluppo, in accordo con lo standard ISO/IEC 12207, è strutturato nel seguente modo:

- Analisi dei requisiti;
- Progettazione;
- Codifica.

### 2.2.4 Implementazione del processo

Si è scelto di adottare un modello di sviluppo *Incrementale*. Il contenuto delle norme sarà aggiornato ad ogni incremento, dando priorità, nei primi incrementi, al soddisfacimento dei requisiti importanti sul piano strategico. I requisiti meno importanti avranno più tempo a disposizione per stabilizzarsi con lo stato del sistema.

### 2.2.5 Analisi dei requisiti

#### 2.2.5.1 Scopo dell'attività

Lo scopo di questa attività è individuare requisiti del progetto descritti nel capitolato presentato e tramite incontri in sede del proponente per concordare requisiti derivati dal dominio tecnologico e applicativo.

#### 2.2.5.2 Aspettative

Il risultato atteso dallo svolgimento di questa attività è la creazione di un documento formale contenente tutti i requisiti concordati e fissati insieme al proponente.

#### 2.2.5.3 Descrizione dell'attività

I requisiti concordati e fissati vanno specificati nel documento *Analisi dei requisiti v1.0.0*. Per analizzare e trovare i requisiti si utilizzano gli use-case. I requisiti vengono tracciati tramite l'applicazione open-source<sub>G</sub> PragmaDB per garantire il controllo di conformità.

#### 2.2.5.4 Strumenti utili al tracciamento

Per il tracciamento dei requisiti e dei casi d'uso<sub>G</sub> si è utilizzata un'applicazione open-source, l'applicazione consiste di un database relazionale<sub>G</sub> a cui è associata un'interfaccia grafica web per facilitarne l'utilizzo. PragmaDB è un applicativo sviluppato da Stefano Munari e Fabio Vedovato. Per maggiori informazioni viene condiviso il link della repository<sub>G</sub> GitHub pubblica dove è possibile consultare la documentazione.

<https://github.com/StefanoMunari/PragmaDB>

#### 2.2.5.5 Casi d'uso

Ogni caso d'uso viene descritto secondo la seguente struttura:

- Codice identificativo;
- Titolo;
- Diagramma Uml<sub>G</sub>;
- Attori primari;
- Attori secondari (se presente);
- Scopo e descrizione;
- Precondizione;

- Postcondizione;
- Scenario principale;
- Scenario alternativo (se presente);
- Estensione (se presente).

#### 2.2.5.6 Codice identificativo

Il codice identificativo descrittivo di ogni caso d'uso è univoco e conforme alla seguente struttura:

**UC(Codice primo livello).(Codice secondo livello).(Codice terzo livello)**

I campi sono rispettivamente:

**Codice di primo livello:** Numero che identifica i casi d'uso da sviluppare;

**Codice di secondo livello:** Numero progressivo che identifica tutti i possibili sotto-casi del caso d'uso enunciato precedentemente;

**Codice di terzo livello:** Numero progressivo che identifica tutti i possibili sotto-casi del sotto-caso in questione.

#### 2.2.5.7 Requisiti

Tutti i requisiti sono descritti secondo la seguente struttura:

- Codice identificativo;
- Tipologia;
- Descrizione;
- Fonti.

#### 2.2.5.8 Codice identificativo

Il codice identificativo descrittivo di ogni requisito è univoco e conforme alla seguente struttura:

**R{X}{Y}{Codice primo livello}.{Codice secondo livello}.{Codice terzo livello}.{Codice quarto livello}**

I campi sono rispettivamente:

- **X:** Identifica il grado di richiesta del proponente
  - **0:** Requisito obbligatorio;
  - **1:** Requisito desiderabile;
  - **2:** Requisito opzionale.
- **Y:** Identifica il tipo di requisito
  - **F:** Requisito funzionale;
  - **Q:** Requisito qualitativo;
  - **V:** Vincolo progettuale.



- **Codice primo livello:** Numero che identifica il requisito da sviluppare;
- **Codice secondo livello:** Numero progressivo che identifica tutti i possibili sotto-casi del requisito enunciato precedentemente;
- **Codice terzo livello:** Numero progressivo che identifica tutti i possibili sotto-casi del sotto-caso enunciato precedentemente;
- **Codice quarto livello:** Numero progressivo che identifica tutti i possibili sotto-casi del sotto-caso enunciato nel terzo livello.

## 3 Processi di supporto

### 3.1 Processo di documentazione

#### 3.1.1 Scopo del processo

Lo scopo di questo processo consiste nell'illustrare come deve essere stilata e gestita la documentazione durante l'intero ciclo di vita del software.

#### 3.1.2 Aspettative

Le aspettative per il suddetto processo si identificano in:

- Fornire una documentazione chiara che venga mantenuta lungo tutto il ciclo di vita del software;
- Norme chiare e semplici per la corretta stesura dei documenti;
- Documentazione formale e coerente.

#### 3.1.3 Descrizione contenuto

Il documento contiene le regole per stilare e mantenere una documentazione corretta e coerente.

#### 3.1.4 Procedura di approvazione dei documenti

Ogni documento non formale deve seguire i seguenti punti:

- Il documento viene stilato dai redattori;
- Il documento viene sottoposto ai *Verificatori* incaricati dal *Responsabile di progetto* per il controllo degli errori;
- Qualora i *Verificatori* trovassero degli errori nella documentazione, devono segnalare i problemi al *Responsabile di progetto* che a sua volta incarica i redattori di apportare le dovute modifiche;
- Se i *Verificatori* ritengono che il documento sia privo di errori, allora il documento viene passato al *Responsabile di progetto* che decide se approvarlo o meno;
- In caso il *Responsabile di progetto* ritenga che il documento non sia ancora in grado di essere approvato, questo deve comunicare una valida motivazione ai redattori con anche le modifiche da apportare al documento.

#### 3.1.5 Template L<sup>A</sup>T<sub>E</sub>X

Per facilitare la creazione di documenti è stato creato un template L<sup>A</sup>T<sub>E</sub>X. Ogni documento costruito in questo modo sarà dotato di una struttura e uno stile di formattazione comune.





### 3.1.6 Struttura dei documenti

#### 3.1.6.1 Costruzione di un documento

I documenti, eccetto i verbali, saranno costruiti a partire da un template  $\text{\LaTeX}$  in cui verranno incluse, tramite il comando `\include`, le sezioni di cui si compone. Le sezioni verranno scritte in file a parte per facilitare il compito dei *Verificatori*.

#### 3.1.6.2 Prima pagina

La prima pagina di ogni documento presenta i seguenti attributi:

- **Logo del gruppo:** Immediatamente visibile come elemento centrato;
- **Titolo del documento:** Immediatamente sottostante al logo;
- **Nome del gruppo e del progetto;**
- **Tabella informativa:** Tabella informativa per la creazione e il versionamento<sub>G</sub> del documento, si compone di:
  - *Versione:* Versione aggiornata del documento;
  - *Data Redazione:* Data di creazione del documento;
  - *Redazione:* Cognome e nome dei redattori del documento;
  - *Verifica:* Cognome e nome dei *Verificatori* del documento;
  - *Approvazione:* Cognome e nome del responsabile<sub>G</sub> di progetto;
  - *Uso:* Interno o esterno;
  - *Distribuzione:* Gruppo che fornisce il determinato documento.

#### 3.1.6.3 Struttura comune delle pagine

Tutte le pagine, ad eccezioni della prima, devono essere costruite rispettando i vincoli adottati nella creazione del template. In ogni pagina devono essere presenti i seguenti elementi:

- Logo del gruppo in alto a sinistra;
- Numero e titolo della sezione corrente in alto a destra;
- Nome del documento e versione attuale in basso a sinistra;
- Pagina corrente su pagine totali indicate come “Pagina di X su Y” dove X indica la pagina corrente e Y la totalità delle pagine di cui si compone il documento.

#### 3.1.6.4 Indice

Ogni documento formale, verbali esclusi, deve essere dotato di un indice, in modo da agevolare la consultazione e permettere una lettura non solo sequenziale. Qualsiasi indice deve partire da 1, per ciascuna sottosezione deve essere presente un punto di separazione dalla sezione padre. Le appendici sono catalogate tramite una lettera maiuscola che verrà incrementata a partire dalla lettera A seguendo poi l'ordine alfabetico internazionale.

### 3.1.6.5 Diario delle modifiche

Ogni documento formale deve contenere questo diario. Questo si compone di una tabella che contiene le modifiche apportate al documento stesso indicando per ognuna:

- Versione del documento;
- Data della modifica;
- Nome e cognome delle persone coinvolte nella sua redazione, verifica e approvazione;
- Ruolo delle persone che hanno redatto, verificato o approvato il documento;
- Descrizione breve e precisa della modifica apportata.

### 3.1.7 Norme tipografiche

#### 3.1.7.1 Elenchi puntati

Gli elenchi puntati sono rappresentati con un pallino nero nel primo livello. Al secondo livello si usa un trattino e un asterisco nel terzo livello. Se è necessario un'ordine, ad ogni livello, gli elementi degli elenchi puntati devono essere rappresentati tramite numeri, che partendo da uno, si incrementano progressivamente. Ogni elenco puntato esprime un concetto, che viene diviso in più parti per facilitare la comprensione.

Ogni elemento di un elenco puntato deve cominciare con la prima lettera maiuscola e concludersi con un punto e virgola, eccetto l'ultimo che mette fine all'elenco con un punto.

#### 3.1.7.2 Comandi personalizzati $\text{\LaTeX}$

Per i riferimenti si utilizzano comandi personalizzati in  $\text{\LaTeX}$  perché garantiscono una corretta scrittura. I comandi sono i seguenti:

- **Ruoli di progetto:**

- $\backslash\text{rdp}$  = *Responsabile di progetto*;
- $\backslash\text{amm}$  = *Amministratore*;
- $\backslash\text{anl}$  = *Analista*;
- $\backslash\text{pro}$  = *Progettista*;
- $\backslash\text{dev}$  = *Programmatore*;
- $\backslash\text{ver}$  = *Verificatore*;
- $\backslash\text{ammi}$  = *Amministratori*;
- $\backslash\text{anli}$  = *Analisti*;
- $\backslash\text{proi}$  = *Progettisti*;
- $\backslash\text{prgi}$  = *Programmatori*;
- $\backslash\text{Verificatori}$  = *Verificatori*.

- **Ruoli di verbale:**

- $\backslash\text{sgr}$  = *Segretario*;

– \sgri = *Segretari*;

• **Documentazione:**

- \pdp = *Piano di progetto*;
- \pdq = *Piano di qualifica*;
- \np = *Norme di progetto*;
- \sdf = *Studio di fattibilità*;
- \adr = *Analisi dei requisiti*;
- \glo = *Glossario*;
- \ddp = *Definizione di prodotto*;
- \mdu = *Manuale utente*.

• **Documentazione compresa di versione:**

- \pdpv = *Piano di progetto v1.0.0*;
- \pdqv = *Piano di qualifica v1.0.0*;
- \npv = *Norme di progetto v1.0.0*;
- \adv = *Analisi dei requisiti v1.0.0*;
- \sdfv = *Studio di fattibilità v1.0.0*;
- \glov = *Glossario v1.0.0*.

• **Revisioni:**

- \revisionedeirequisiti = **Revisione dei Requisiti**;
- \revisionediprogettazione = **Revisione di Progettazione**;
- \revisionediquifica = **Revisione di Qualifica**;
- \revisionediaccettazione = **Revisione di Accettazione**.

• **Nome del team:**

- \wm = *WarMachine*.

• **Nome del proponente:**

- \proponente = *Zucchetti s.p.a.*

• **Nome del committente:**

- \committente = *Prof. Vardanega Tullio e Prof. Cardin Riccardo* .

• **Nome di progetto:**

- \progetto = *IronWorks*.

- **Nuova riga di Diario delle modifiche:** Il comando \diariomodifichenew permette di aggiungere una riga alla tabella del diario modifiche di un documento.



### 3.1.7.3 Sigle e abbreviazioni

Nella documentazione si utilizzano sigle e abbreviazioni per l'uso di tabelle o altre strutture con spazio limitato. Le sigle sono le seguenti:

- Per i nominativi della documentazione:
  - **RR**: Revisione dei requisiti;
  - **RP**: Revisione di progettazione;
  - **RQ**: Revisione di qualifica;
  - **RA**: Revisione di accettazione;
  - **NP**: Norme di Progetto;
  - **PP**: Piano di progetto;
  - **PQ**: Piano di qualifica;
  - **AR**: Analisi dei requisiti;
  - **SF**: Studio di fattibilità;
  - **VB**: Verbale;
  - **GL**: Glossario.
- Per i nominativi dei rischi alla sezione analisi dei rischi sul *Piano di progetto*:
  - **ITT**: Inesperienza tecnologica del Team;
  - **GHS**: Guasti e problemi hardware e software;
  - **PD**: Perdita dati;
  - **PC**: Problemi di calendario;
  - **IIM**: Indisponibilità improvvisa dei membri;
  - **ILT**: Inesperienza lavorativa del Team;
  - **ROR**: Rotazione ruoli;
  - **IUS**: Inesperienza del Team nell'uso degli strumenti;
  - **CR**: Comprensione requisiti;
  - **SCA**: Sottovalutazione costi attività.
- Per i nominativi dei ruoli di progetto:
  - **AM**: Amministratore;
  - **RE**: Responsabile di progetto;
  - **AN**: Analista;
  - **PT**: Progettista;
  - **PM**: Programmatore;
  - **VR**: Verificatore.



#### 3.1.7.4 Stile del testo

- **Glossario:** Ogni parola contenuta nel Glossario viene marcata con una G maiuscola nel pedice: per esempio committente. Viene segnalata soltanto la prima occorrenza del termine presente nel glossario;
- **Grassetto:** Viene applicato ai titoli delle sezioni e alle parole chiave che compongono il primo livello di un elenco puntato;
- **Corsivo:** Il corsivo viene utilizzato nei seguenti casi:
  - Secondo livello degli elenchi puntati;
  - Abbreviazioni;
  - Parole facenti parte del glossario;
  - Nomi di documentazione;
  - Nomi di azienda o titoli professionali;
  - Ruoli di progetto;
  - Nomi delle parti coinvolte nel progetto.
- **Monospace:** Viene utilizzato per i nomi dei file;

#### 3.1.7.5 Formati

I formati utilizzati nella documentazione sono:

- **Orario:** hh:mm  
Dove hh indica le ore e varia tra 0 e 23, mentre mm indica i minuti e varia tra 0 e 59;
- **Data:** gg/mm/yyyy  
Dove gg rappresenta il giorno utilizzando due cifre al massimo, mm rappresenta i mesi utilizzando al massimo due cifre e yyyy rappresenta gli anni utilizzando quattro cifre.
- **Nomi di rilievo:**
  - **Nome di ruolo di progetto:** Viene scritto con la prima lettera maiuscola e in stile corsivo;
  - **Nome delle parti coinvolte nel progetto:** Rientrano in questa categoria fornitore, proponente e committente. Questi nomi sono scritti con la prima lettera maiuscola e in stile corsivo;
  - **Nome di documentazione:** Viene scritto con la prima lettera maiuscola e in stile corsivo;
  - **Nome di file:** Viene scritto senza spaziatura;
  - **Nome proprio di persona:** Viene scritto con la prima lettera di nome e cognome maiuscola, nell'ordine Cognome Nome.

### 3.1.8 Componenti grafiche

#### 3.1.8.1 Tabelle

Ogni tabella deve essere centrata orizzontalmente e avere, sotto di essa, la propria didascalia associata in modo tale da renderla di più semplice comprensione. Nella didascalia andrà inserito il numero identificativo incrementale per il tracciamento associato a una breve descrizione del contenuto.

#### 3.1.8.2 Immagini

Le immagini devono essere centrate orizzontalmente e avere una larghezza variabile in base alla dimensione originale dell'immagine.

Per agevolare la consultazione della documentazione le immagini verranno separate verticalmente dal testo. L'immagine dovrà essere associata a una didascalia, composta di un numero identificativo utile per il tracciamento all'interno del documento e una breve descrizione di cosa essa rappresenta. I diagrammi Uml vengono inseriti nella documentazione come immagini.

### 3.1.9 Classificazione dei documenti

#### 3.1.9.1 Documenti informali

I documenti che rientrano in questa categoria sono tutti quelli che non sono ancora stati approvati dal *Responsabile di progetto*, sono da considerarsi esclusivamente ad uso interno fino ad approvazione.

#### 3.1.9.2 Documenti formali

I documenti che rientrano in questa categoria sono tutti quelli che hanno ricevuto approvazione dal *Responsabile di progetto*. Questi documenti sono da considerarsi per uso esterno, saranno i documenti condivisi con committente e proponente. Per arrivare a tale stato il documento deve aver già passato la fase di verifica e validazione.

#### 3.1.9.3 Verbali

Il verbale è un documento non formale redatto da un *Segretario* in occasione di incontri con il proponente o ritrovi dei membri del gruppo per discutere problematiche relative al progetto.

Il verbale è un documento non soggetto a versionamento, viene redatto una sola volta e non subisce successive modifiche. Il verbale dovrà essere approvato dal *Responsabile di progetto*. Ogni verbale dovrà indicare nel seguente ordine le sezioni:

1. "Informazioni generali":

- **Luogo incontro:** Città (Provincia), Via, Sede;
- **Data incontro:** gg-mm-yyyy;
- **Ora inizio incontro:** hh-mm formato 24h;
- **Ora fine incontro:** hh-mm formato 24h;
- **Durata incontro:** Indicato in ore;
- **Oggetto<sub>G</sub>:** Argomenti trattati;

- **Segretario:** Cognome Nome;
- **Partecipanti:** Cognome Nome.

2. **"Riassunto dell'incontro":** contenente le decisioni prese, descritte come elementi di un elenco puntato;
3. **"Riepilogo":** Contenente una tabella che riprende i codici identificativi e le decisioni relative ad essi.

### 3.1.10 Strumenti

#### 3.1.10.1 L<sup>A</sup>T<sub>E</sub>X

Per la stesura della documentazione è stato scelto di usare il linguaggio L<sup>A</sup>T<sub>E</sub>X. La scelta è stata dettata da questioni di praticità, in quanto è utile per separare il contenuto dalla formattazione permettendo un versionamento agevolato e la creazione di template per l'impaginazione condivisa dei documenti. Inoltre il linguaggio permette una comoda gestione automatica degli indici e dei glossari.

#### 3.1.10.2 TexMaker

Per la scrittura del codice L<sup>A</sup>T<sub>E</sub>X si è utilizzato TexMaker, di versione minima 4.5. L'applicazione offre un pacchetto con compilatore e visualizzatore PDF integrato e fornisce suggerimenti sul completamento dei comandi L<sup>A</sup>T<sub>E</sub>X.

<http://www.xm1math.net/texmaker/>

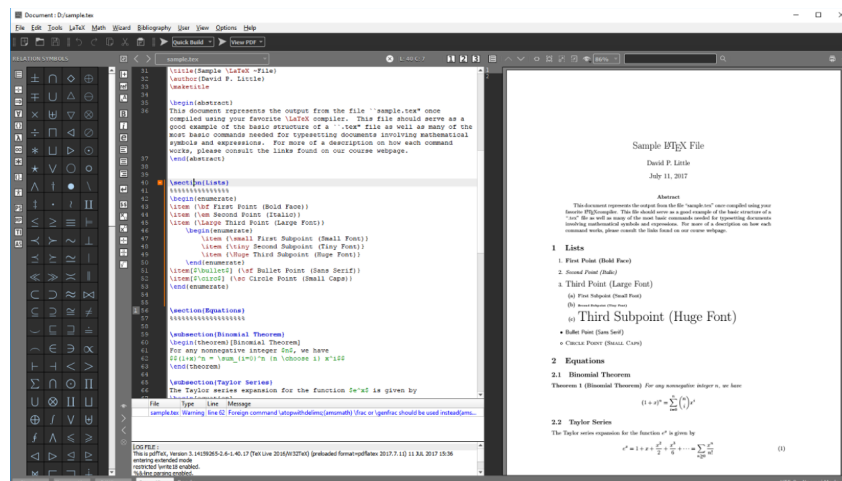


Figura 1: Immagine esempio TexMaker.

#### 3.1.10.3 Astah

Per la realizzazione dei diagrammi degli Use-Case ricavati nel documento *Analisi dei requisiti* si è scelto di utilizzare Astah. Astah è un' applicazione che fornisce un'interfaccia utente che comprende un editor<sub>G</sub> per diagrammi UML e dei tool da utilizzare per la modellazione del diagramma. L'applicazione consiste in una versione gratuita e di una versione premium. Si è deciso di utilizzare la versione premium poiché offriva più possibilità per la modellazione.

<http://astah.net/>

#### 3.1.10.4 Lucidchart

Per la realizzazione dei diagrammi a scopo illustrativo utilizzati nella documentazione si è utilizzato Lucidchart, un' applicazione web<sub>G</sub> che permette di realizzare diversi tipi di diagrammi.

<https://www.lucidchart.com>

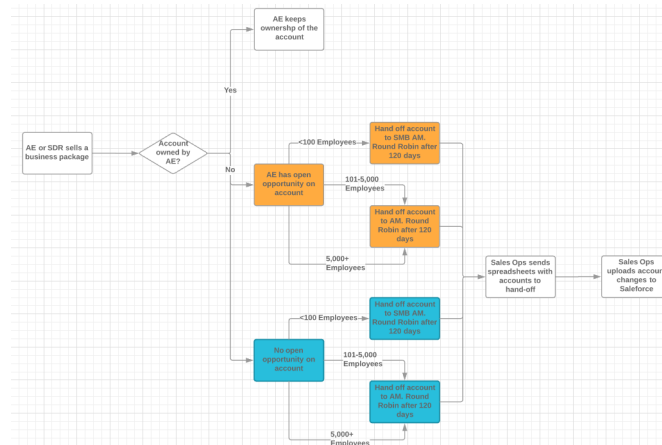


Figura 2: Immagine esempio Lucidchart.

#### 3.1.10.5 Microsoft Excel

Per calcolare il preventivo da proporre, la realizzazione di grafici e tabelle per la gestione del tempo rendicontato e dell'impatto percentuale dei costi rispetto ai vari ruoli è stato usato Microsoft Excel.

### 3.2 Processo di gestione delle versioni e delle configurazioni

#### 3.2.1 Versionamento della documentazione

Ciascun documento deve essere soggetto a versionamento, eccetto i verbali, perciò chiunque lo utilizzi avrà una visione chiara e dettagliata della sua storia e delle sue modifiche. Ad ogni versione deve corrispondere una riga nel diario delle modifiche. Verrà applicato il seguente formalismo:

vX.Y.Z

dove:

- **v**: Versione del documento;
- **X**:
  - Inizia da 0;
  - Viene incrementato quando il Responsabile di Progetto approva il documento;



- Limitato superiormente dal numero di revisioni.
- **Y:**
  - Inizia da 0;
  - Viene incrementato da parte del Verificatore ad ogni verifica;
  - Non è limitato superiormente;
  - Quando viene incrementato X, Y e Z vengono riportati a 0.
- **Z:**
  - Inizia da 0;
  - Viene incrementato da parte del Redattore del documento ad ogni modifica;
  - Non è limitato superiormente;
  - Quando viene incrementato Y, viene riportato a 0.

### 3.2.2 Gestione del versionamento

#### 3.2.2.1 Repository

Per il versionamento e il salvataggio dei file prodotti durante il progetto è previsto l'utilizzo del repository creato su gitHub. L' *Amministratore* di Progetto si occupa di creare tali repository con un account generale, rappresentante il gruppo di progetto chiamato "WarMachineSwe". Successivamente, tutti gli altri membri del gruppo dovranno creare un account personale, che sarà autorizzato all'accesso dei repository dall' *Amministratore*.

#### 3.2.2.2 Struttura del repository

La struttura del repository è composta da quattro cartelle, una per ogni tipo di revisione (RR, RQ, RP e RA). A loro volta, queste si sviluppano nelle cartelle seguenti:

- **interni:** Contenente tutti i documenti a scopo interno;
- **esterni:** Contenente tutti i documenti a scopo esterno;
- **textemplates:** Contiene i file del template e il layout dei documenti  $\text{\LaTeX}$ ;
- **scripts:** Contiene gli scripts scritti nel linguaggio di programmazione Python creati per automatizzare determinati compiti.

#### 3.2.2.3 Tipi di file e .gitignore

Nelle cartelle contenenti tutti i documenti saranno presenti solamente i file .tex, .pdf, .jpg, .png. Le estensioni dei file generati automaticamente dalla compilazione sono stati aggiunti al file .gitignore nel repository, quindi vengono ignorati e resi invisibili a git.

#### 3.2.2.4 Norme sui commit

Ogni qual volta che vengono effettuate delle modifiche ai file del repository, e successivamente devono essere caricate su di esso, bisogna specificarne le motivazioni nel modo più chiaro possibile. Questo avviene utilizzando il comando "commit" accompagnato da un messaggio riassuntivo e una descrizione in cui va specificato:

- La lista dei file coinvolti;
- La liste delle modifiche effettuate, ordinate per ogni singolo file.

Prima di eseguire tale procedura, va aggiornato il diario delle modifiche, secondo le regole viste nella sezione 3.1.6.5.

### 3.3 Processo di verifica

#### 3.3.1 Scopo del processo

Lo scopo di questo processo è quello di accertare che l'esecuzione di specifiche attività non abbia introdotto errori.

#### 3.3.2 Aspettative

Dall'esecuzione di questo processo è possibile individuare:

- Una o più procedure di verifica;
- Regole per verificare la correttezza del prodotto;
- Imprecisioni, errori e imperfezioni del prodotto, i quali vengono annotati per essere corretti.

#### 3.3.3 Descrizione del processo

Il processo è diviso in due attività:

- **Analisi:** Questa attività viene effettuata in due modi, l'analisi statica e l'analisi dinamica.
- **Test di verifica:** Consiste in test definiti dal team *WarMachine* per verificare la corretta esecuzione del prodotto. Dal momento che ancora non si sono trattate le tecnologie richieste questa parte sarà posticipata al completamento della revisione dei requisiti.

#### 3.3.4 Linee guida per la verifica della documentazione

Per la verifica dei documenti sono fornite le seguenti linee guida che i *Verificatori* devono seguire per garantire un corretto processo di verifica. Le linee guida sono:

- Verificare che siano presenti tutte le sezioni indicate nell'indice del documento;
- Verificare la grammatica utilizzata e la sintassi del testo del documento;

- Verificare la grammatica nel testo associato alle figure e alle tabelle (captions, testo interno);
- Verificare che il testo del documento non contenga termini qualitativi ambigui. Es. Abbiamo svolto un'analisi approfondita, "approfondita" sarebbe meglio non usarlo a meno che non sia stata definita una misura precisa;
- Verificare che le frasi siano di senso compiuto;
- Verificare che i periodi non risultino inutilmente lunghi;
- Verificare che il soggetto e l'oggetto dell'azione siano sempre indicati;
- Verificare che i termini ambigui o di difficile comprensione siano trattati nel glossario.

### 3.3.5 Metodi di segnalazione degli errori

I metodi usati dal gruppo per segnalare che un *Verificatore* ha trovato degli errori sono due:

- **Modifiche al diario modifiche:** I *Verificatori* quando avranno finito il processo di verifica registreranno i risultati sul diario delle modifiche, considerando sia le parti verificate con successo sia le parti che hanno bisogno di essere riviste. Per quanto riguarda le parti verificate correttamente, per facilitare la scrittura e la comprensione, si scrivono solo le sezioni verificate segnalando gli errori nella maniera più precisa possibile. Le segnalazioni devono essere scritte nel seguente modo:

Da rivedere [W].[X].[Y].[Z]

Dove ogni lettera, rispettivamente, indica un diverso livello di annidamento:

- *W*: La sezione di riferimento;
- *X*: La sotto-sezione di riferimento;
- *Y*: La sotto-sotto-sezione di riferimento;
- *Z*: Il paragrafo di riferimento.

Qualora si presentino più errori nello stesso livello di annidamento si deve segnalare l'intero livello.

- **GitHub Issues:** Per la gestione degli errori interna e il loro tracciamento viene utilizzato il processo di segnalazione di errori di GitHub. Lo strumento verrà descritto nel dettaglio al paragrafo 4.1.9.5.

#### 3.3.5.1 Template GitHub Issues per segnalazione degli errori

Per la segnalazione di errori utilizzando GitHub Issues è stato creato un template. Il template ha la funzione di fissare una struttura comune e di facilitare la comprensione per le segnalazioni.

### 3.3.6 Analisi

#### 3.3.6.1 Analisi statica

L'analisi statica è un metodo di verifica del codice sorgente e della documentazione. Non richiede l'esecuzione di parti del sistema software e viene eseguita tramite una delle seguenti tecniche:

- **Walkthrough:** Si tratta di una lettura a largo spettro. Il suo scopo è di esaminare l'intero documento alla ricerca di errori generici. Non è una tecnica di analisi molto pratica, viene usata principalmente nelle prime fasi<sub>G</sub> di progetto quando il gruppo non ha ancora buona padronanza o conoscenza delle norme e del *Piano di qualifica*. A supporto di questa tecnica si deve creare una lista di controllo dove vengono elencati gli errori più comuni.
- **Inspection:** Si tratta di una lettura mirata e strutturata, si applica quando si conoscono le problematiche per cui è richiesta la verifica. Questa tecnica viene eseguita da *Verificatori* scelti e distinti dai programmatori che, durante la lettura, cercheranno di individuare gli errori segnalati nella lista di controllo precedentemente stilata. Con l'avanzare nel progetto la lista di controllo viene estesa rendendo questa tecnica ancor più efficace.

#### 3.3.6.2 Analisi dinamica

E' una tecnica di analisi del prodotto software, comporta l'esecuzione del programma. La tecnica viene applicata tramite dei test che hanno la funzione di verificare il corretto funzionamento del prodotto oppure di far emergere eventuali errori in esso. I test sono procedure costose, richiedono tempo, persone e infrastrutture. Un singolo test non basta, i risultati valgono solo per quella determinata esecuzione ma non valgono più a valle di una nuova modifica. I test devono essere ripetibili cioè deve essere possibile, dato lo stesso input e svolto nello stesso ambiente, ottenere lo stesso output. I test sono caratterizzati come segue:

- **Ambiente:** Rappresenta il sistema hardware e software dove viene fatto eseguire il test del prodotto;
- **Stato iniziale:** Rappresenta il punto iniziale da cui far eseguire il test;
- **Ingresso:** Rappresenta l'input inserito;
- **Uscita:** Rappresenta l'output ottenuto;
- **Oggetto della prova:** Rappresenta l'oggetto che viene sottoposto al test di verifica;

### 3.3.7 Test di verifica

#### 3.3.7.1 Test di unità[TU]

Il test consiste nell'isolare singole unità di software per testare se la parte isolata funziona come previsto. Per unità si intende la parte di software più piccola verificabile. A seconda del paradigma di programmazione adottato può identificarsi con una funzione o una classe, rispettivamente se il paradigma è procedurale o ad oggetti. Le unità vengono sottoposte

al test prima di essere integrate in moduli ed essere sottoposte al test delle interfacce tra i vari moduli. Per i test di unità sono richieste le componenti  $\text{stub}_G$  e  $\text{driver}_G$ . I driver simulano un'unità chiamante mentre lo stub simula l'unità chiamata.

### 3.3.7.2 Test di integrazione[TI]

Il test consiste nella combinazione di unità, le quali devono essere già state sottoposte al test di unità, per formare un unico componente e nel test delle interfacce effettuato successivamente per valutare se l'interazione tra le parti combinate presenta problemi. Le strategie di test di integrazione possono essere di due tipi:

- **Bottom-up:** Prima vengono sviluppate le parti con minori dipendenze funzionali e di maggiore utilità. Successivamente si prosegue risalendo l'albero delle dipendenze. Questa strategia riduce il numero di stub ma non rende immediatamente disponibili funzionalità di alto livello;
- **Top-down:** Vengono prima sviluppate le parti più esterne e successivamente si scende lungo l'albero delle dipendenze. Questa strategia aumenta il numero di stub ma rende presto disponibili funzionalità di alto livello.

### 3.3.7.3 Test di sistema[TS]

Il test di sistema viene svolto dopo il test di integrazione ed è effettuato sul prodotto software finale, ovvero quando lo si ritiene pronto per la consegna. Il test consiste nella verifica del soddisfacimento dei requisiti da parte del prodotto software.

### 3.3.7.4 Test di regressione[TR]

Il test va eseguito quando vengono modificate parti del sistema, la sua funzione è accertare che modifiche effettuate per correzione o estensione non comportino errori nel sistema. Per verificare che non siano emersi problemi dalle parti verificate si devono eseguire nuovamente i test TU e TI sulla parte di codice che ha ricevuto modifiche.

### 3.3.7.5 Test di accettazione[TA]

Il test consiste nel collaudo del prodotto in presenza del proponente. Una volta superato il test si prosegue con il rilascio ufficiale del prodotto.

## 3.3.8 Strumenti

### 3.3.8.1 Verifica ortografica

Viene utilizzata la verifica dell'ortografia in tempo reale, strumento integrato in TexMaker che sottolinea in rosso le parole errate secondo il dizionario della lingua italiana.

### 3.3.8.2 GitHub Issues

Per l'assegnazione e il tracciamento degli errori viene utilizzato lo strumento issues di GitHub con linguaggio markdown<sub>G</sub>. Gli issue sono composti da elenchi puntati i cui elementi sono righe di codice che necessitano di essere riviste. Per ogni issue verrà il *Responsabile di progetto* assegna un membro del gruppo che si occupa della sua risoluzione.

<https://github.com/WarMachineSwe/WarMachine/issues>

### 3.3.8.3 Travis CI

Travis CI è un servizio distribuito di integrazione continua<sub>G</sub> che permette di eseguire script<sub>G</sub> e test automatizzati<sub>G</sub> quando avvengono modifiche nella repository.

Ad ogni push<sub>G</sub>, Travis CI esegue gli scripts specificati e i test automatizzati, il tutto in un ambiente di sviluppo vergine<sub>G</sub> garantendo l'effettivo successo della build<sub>G</sub> del software. Tramite uno script, Travis, calcola l'indice di gulpease<sub>G</sub>. Inoltre, deve essere usato anche come strumento di correttezza per il glossario. L'applicazione permette infatti le seguenti verifiche:

- Segnala come errore i termini, trovati nel testo, marcati col comando \gl ma non presenti all'interno del glossario;
- Segnala come errore i termini, trovati nel glossario, che non sono presenti in nessuna sezione del testo.

<https://travis-ci.org/WarMachineSwe/WarMachine>

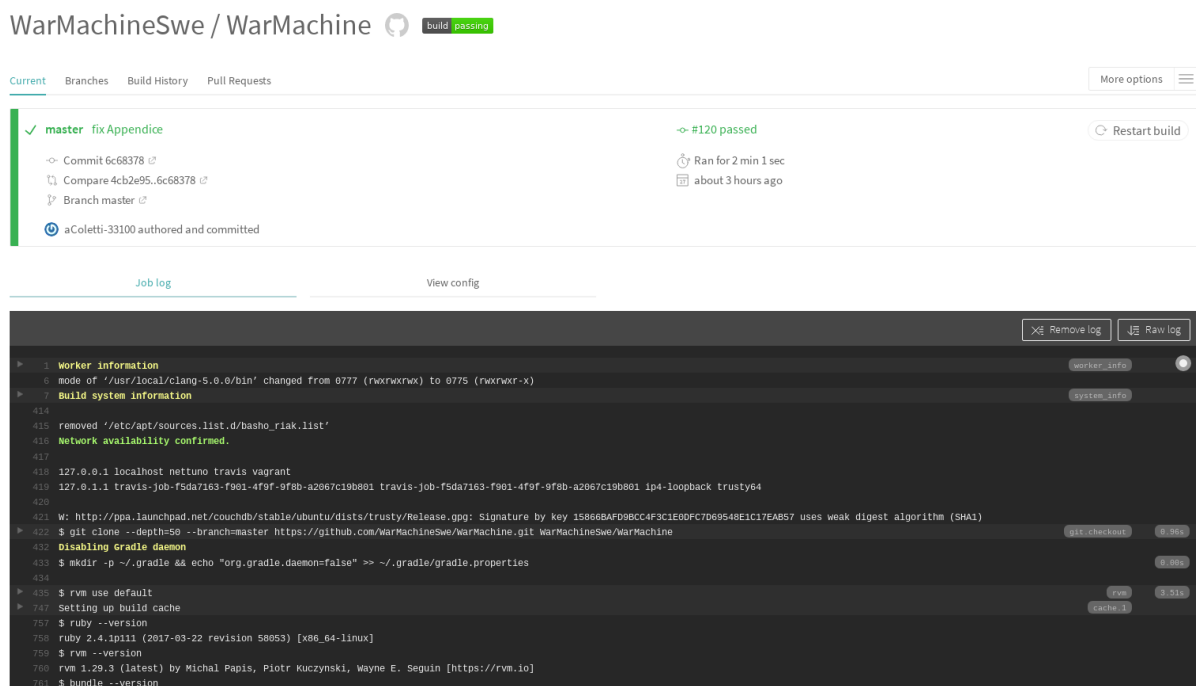


Figura 3: Immagine che illustra una delle funzionalità di Travis CI.

### 3.3.9 Metriche

Il processo di verifica, per essere informativo, dovrà innanzitutto essere quantificabile. I dati ottenuti attraverso il processo di verifica dovranno basarsi su metriche stabilite a priori e qualora queste fossero ambigue e approssimate si miglioreranno attraverso il ciclo di vita, in maniera incrementale. Data la natura variabile di tali metriche si sono individuati due range:

- **Accettazione:** Valore richiesto per accettare il prodotto;
- **Ottimale:** Valore entro cui dovrebbe inserirsi la misurazione. Non è vincolante, ma fortemente consigliato e uno scostamento da questo necessita di ulteriori verifiche.

### 3.3.10 Metriche per i processi

Queste metriche utilizzano indici che ne analizzano l'andamento di variabili critiche come costi e tempi. Tali indici forniscono informazioni che consentono un riscontro immediato sullo stato attuale del progetto; ad ogni incremento questi verranno valutati e, in base ai risultati, il responsabile di progetto sceglierà come procedere.

#### 3.3.10.1 Modello Spice<sub>G</sub>

Per gestire e valutare i processi si deve usare il modello SPICE.

Effettuando questo tipo di verifiche il team avrà subito un riscontro della qualità<sub>G</sub> del processo. Lo SPICE contiene un modello di riferimento, questo definisce una dimensione di processo e una dimensione di capacità. La dimensione di processo divide i processi nelle seguenti categorie:

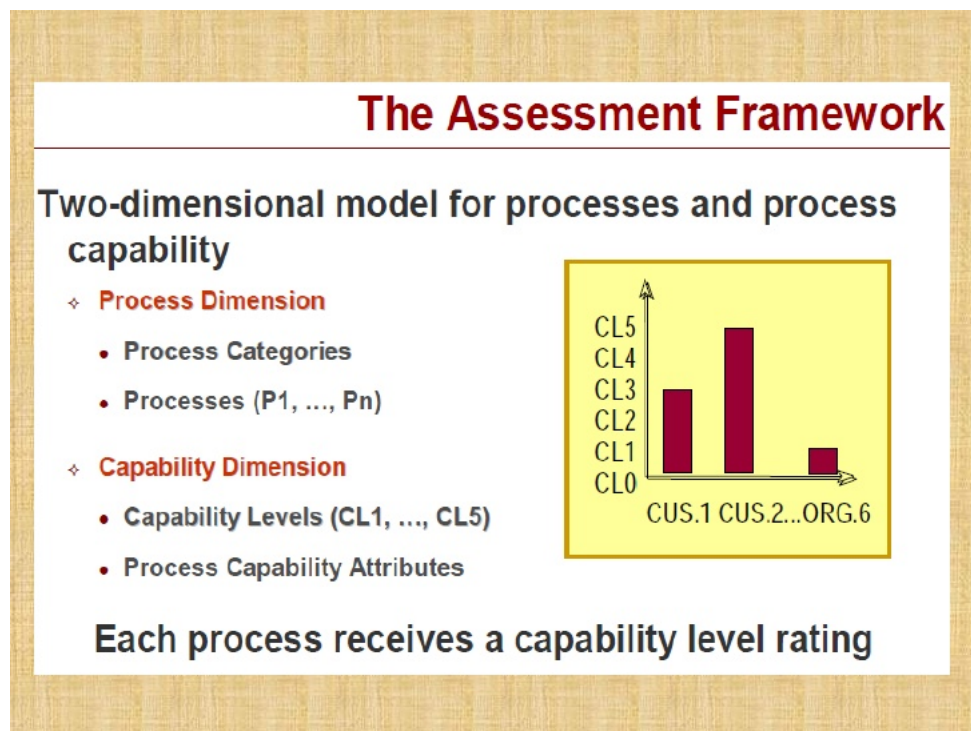
- **Customer/Supplier;**
- **Engineering;**
- **Supporting;**
- **Management;**
- **Organization.**

Ad ogni processo viene assegnato un livello della seguente scala:

- 0: processo incompleto;
- 1: processo eseguito;
- 2: processo gestito;
- 3: processo stabilito;
- 4: processo prevedibile;
- 5: processo completo.

La capacità dei processi è misurata tramite i seguenti attributi definiti a livello internazionale:

- 1.1 Process Performance;
- 2.1 Performance Management;
- 2.2 Work Product Management;
- 3.1 Process Definition;
- 3.2 Process Deployment;
- 4.1 Process Measurement;
- 4.2 Process Control;
- 5.1 Process Innovation;
- 5.2 Process Optimization.



**Figura 4:** Immagine esplicativa per ISO/IEC 15504.

Ciascun attributo di processo consiste di una o più pratiche generiche che a loro volta sono elaborate in "Indicatori della pratica" che aiutano nella fase di valutazione delle prestazioni.

Ciascun attributo del processo è valutato secondo una scala a quattro valori (N-P-L-F):

- Not achieved (0 - 15%);
- Partially achieved (>15% - 50%);
- Largely achieved (>50% - 85%);



- **Fully achieved** (>85% - 100%).

La valutazione è fatta sulla base di evidenze oggettive acquisite a fronte di ciascun indicatore di processo durante la fase di assessment che dimostrano l'aderenza agli attributi dei processi.

### 3.3.10.2 Ciclo Di Deming

Per il miglioramento continuo dei processi e dei prodotti si è adottato il ciclo di Deming. Questo metodo è composto dalle seguenti quattro fasi:

- **Plan:** Stabilire gli obiettivi e i processi necessari per fornire risultati in accordo con i risultati attesi, attraverso la creazione di attese di produzione, di completezza e accuratezza delle specifiche scelte. Quando possibile, avvio su piccola scala, per verificare i possibili effetti;
- **Do:** Esecuzione del programma, dapprima in contesti circoscritti. Attuare il piano, eseguire il processo, creare il prodotto. Raccogliere i dati per la creazione di grafici e analisi da destinare alla fase di "Check" e "Act";
- **Check:** Test e controllo, studio e raccolta dei risultati e dei riscontri. Studiare i risultati, misurati e raccolti nella fase del "Do" confrontandoli con i risultati attesi, obiettivi del "Plan", per verificarne le eventuali differenze. Cercare le deviazioni nell'attuazione del piano e focalizzarsi sulla sua adeguatezza e completezza per consentirne l'esecuzione. I grafici dei dati possono rendere questo molto più facile, in quanto è possibile vedere le tendenze di più cicli PDCA, convertendo i dati raccolti in informazioni. L'informazione è utile per realizzare il passo successivo : "Act";
- **Act:** Azione per rendere definitivo e/o migliorare il processo (estendere quanto testato dapprima in contesti circoscritti all'intera organizzazione). Richiede azioni correttive sulle differenze significative tra i risultati effettivi e previsti. Analizza le differenze per determinarne le cause e dove applicare le modifiche per ottenere il miglioramento del processo o del prodotto. Quando un procedimento, attraverso questi quattro passaggi, non comporta la necessità di migliorare la portata a cui è applicato, il ciclo PDCA può essere raffinato per pianificare e migliorare con maggiore dettaglio la successiva iterazione, oppure l'attenzione deve essere posta in una diversa fase del processo.

### 3.3.10.3 SV - Schedule Variance<sub>G</sub>

La Schedule Variance indica l'efficacia dei processi valutando l'andamento temporale delle attività che li compongono: vale a dire se la conclusione delle attività è in regola, in ritardo o concordante con i tempi prefissati nel documento di *Piano di progetto*.

Valori utilizzati nella formula:

- **DCR** : Valore (in giorni) delle attività realizzate alla data corrente;
- **DCP** : Costo pianificato (in giorni) per realizzare le attività di progetto alla data corrente.

$$SV = DCR - DCP \quad (1)$$

I risultati ottenibili sono:

- **Valore non accettabile:** Un valore maggiore o uguale al 10% del costo totale prefissato;
- **Valore accettabile:** Un valore minore del 10% del costo totale prefissato;
- **Valore ottimale:** Un valore pari allo 0%, indicatore di una pianificazione accurata e ben gestita.

#### 3.3.10.4 CV - Cost Variance<sub>G</sub>

E' un indicatore di efficienza. Il costo reale si deve confrontare rispetto al costo prefissato nel documento *Piano di progetto*. In caso il costo reale fosse maggiore del costo prefissato questo indica che sono state richieste risorse maggiori rispetto a quanto previsto precedentemente.

Valori utilizzati nella formula:

- **CR** :Valore (in €) delle attività realizzate alla data corrente;
- **CP** : Costo pianificato (in €) per realizzare le attività di progetto alla data corrente.

$$CV = CR - CP \quad (2)$$

I risultati ottenibili sono:

- **Valore non accettabile:** Un valore maggiore o uguale al 10% del costo totale prefissato;
- **Valore accettabile:** Un valore minore del 10% del costo totale prefissato;
- **Valore ottimale:** Un valore vicino, pari o inferiore allo 0%, indicatore di una pianificazione accurata e ben gestita.

#### 3.3.11 Metriche per i documenti

Per quanto riguarda i documenti, la metrica scelta è un indice di leggibilità basato sulla lingua italiana, gulpease.

### 3.3.11.1 Indice di Gulpease

Permette di indicare la leggibilità di un testo attraverso un valore compreso fra zero e cento ed è l'indice più diffuso data la sua semplicità d'uso.

L'indice Gulpease considera due variabili linguistiche: la lunghezza della parola e la lunghezza della frase rispetto al numero delle lettere ed è calcolato attraverso la seguente formula:

$$89 + \frac{300 \times (\text{numero delle frasi}) - 10 \times (\text{numero delle lettere})}{\text{numero delle parole}}$$

Più alto sarà il valore ottenuto da questa formula, maggiore sarà la leggibilità del documento e viceversa.

In generale è risultato che i testi con un indice:

- [**<80**]: Sono difficili da leggere per chi ha la sola licenza elementare;
- [**<60**]: Sono difficili da leggere per chi ha la sola licenza media;
- [**<40**]: Sono difficili da leggere per chi ha la licenza superiore.

### 3.3.11.2 Indice di errori concettuali

Indica se un documento è corretto rispetto al suo contenuto, in altre parole se i suoi concetti sono coerenti e corretti lungo tutto il documento. L'indice ottenuto con questa metrica rappresenta quanti errori concettuali non sono stati corretti dopo esser stati segnalati da un *Verificatore*. Variabili utilizzate nella formula:

- **enc** = Numero di errori non ancora corretti;
- **ec** = Numero totale degli errori segnalati.

La metrica si calcola con la seguente formula:

$$Errori = \frac{enc}{ec} * 100 \quad (3)$$

I risultati ottenibili sono:

- [**>5**]: Valore non sostenibile;
- [**<=5**]: Valore sostenibile;
- [**0**]: Valore ottimale.



## 4 Processi organizzativi

### 4.1 Processo di gestione di progetto

#### 4.1.1 Scopo

Lo scopo dei processi in questione è la produzione del documento *Piano di progetto*, rivolto a stabilire e implementare l'organizzazione e la definizione dei ruoli di ogni membro del team.

#### 4.1.2 Aspettative

Le aspettative di tali processi sono:

- produzione del Piano di Progetto;
- definizione dei ruoli di progetto;
- definizione delle modalità di esecuzione dei compiti.

#### 4.1.3 Descrizione

Orario di lavoro:

- dalle 9.30 alle 12.30;
- dalle 14.30 alle 18.30.

Viene trattata la gestione dei seguenti argomenti:

- Ruoli di progetto;
- Comunicazioni;
- Incontri;
- Strumenti di coordinamento;
- Gestione dei rischi.

#### 4.1.4 Ruoli di progetto

Lungo l'intero ciclo di vita del progetto, sono previsti diversi ruoli e si è scelto di farli ricoprire almeno una volta ad ogni membro. Nel *Piano di progetto v1.0.0* vengono pianificate le attività assegnate agli specifici ruoli di progetto. Le attività e i compiti per ogni ruolo sono brevemente spiegati come segue.



#### 4.1.4.1 Amministratore di progetto

Il compito principale di chi ricopre il ruolo di *Amministratore* di progetto è controllare ed amministrare l'ambiente di lavoro, avendo diretta responsabilità sulla capacità operativa e sull'efficienza.

Le responsabilità assunte sono:

- Studio e ricerca di strumenti da fornire ai membri del gruppo che migliorino l'ambiente di lavoro, automatizzando dove possibile;
- Controllare versioni e configurazioni del prodotto software;
- Controllo della qualità sul prodotto, fornendo procedure e strumenti di monitoraggio o segnalazione;
- Risolvere i problemi legati alla gestione dei processi e delle risorse disponibili;
- Gestire il versionamento e l'archiviazione di tutta la documentazione di progetto.

#### 4.1.4.2 Responsabile di progetto

Il *Responsabile di progetto* rappresenta il punto di riferimento del team, per il committente e il fornitore. Egli approva le scelte prese dal gruppo e se ne assume la responsabilità.

Le responsabilità assunte sono:

- Approvazione della documentazione;
- Approvazione dell'offerta economica;
- Gestione delle risorse umane;
- Coordinamento e pianificazione delle attività di progetto;
- Studio e gestione rischi.

#### 4.1.4.3 Analista

L'*Analista* il compito di effettuare un'analisi approfondita sul dominio del problema, in modo da apprenderlo in maniera esaustiva. Non è necessaria la sua presenza durante l'intera durata del progetto.

Le responsabilità assunte sono:

- Comprensione della natura del problema e della sua complessità;
- Produzione dello Studio di Fattibilità e dell'Analisi dei Requisiti, delineando delle specifiche il più possibile comprensibili, sia dal proponente che dal committente.

#### 4.1.4.4 Progettista

Il *Progettista* deve avere una conoscenza approfondita delle tecnologie utilizzate e competenze tecniche aggiornate, in modo da poter gestire gli aspetti tecnici e tecnologici del progetto.

Le responsabilità assunte sono:

- Effettuare scelte su aspetti tecnici del progetto, il più possibile efficienti ed ottimizzate;
- Effettuare scelte per rendere il prodotto facilmente mantenibile.

#### 4.1.4.5 Verificatore

Il *Verificatore* deve avere un'ampia conoscenza delle normative del progetto, in modo da poterne garantire una corretta verifica.

La responsabilità assunta è:

- Controllare che le attività di progetto siano conformi alle norme stabilite.

#### 4.1.4.6 Programmatore

Il *Programmatore* è responsabile delle attività di codifica e di creazione delle componenti di supporto, utili a effettuare le prove di verifica e validazione sul prodotto software.

Le responsabilità assunte sono:

- Implementare le soluzioni previste dal Progettista;
- Scrivere codice pulito e conforme alle norme di progetto, in modo da essere facilmente mantenibile;
- Versionare il codice prodotto;
- Realizzare gli strumenti utili per poter compiere le prove di verifica e validazione previste.

#### 4.1.5 Gestione delle comunicazioni

La sezione in questione espone le norme da rispettare per le comunicazioni tra le varie parti (proponente, committenti, componenti del gruppo).

##### 4.1.5.1 Comunicazioni interne

Per le comunicazioni interne tra membri del gruppo *WarMachine* viene utilizzato un software denominato  $\text{slack}_G$ , uno dei maggiori strumenti di messaggistica per la collaborazione aziendale. Questo servizio permette la creazione di canali specifici per una determinata attività, in cui è possibile lo scambio di informazioni e la condivisione di file che potranno essere accessibili a tutto il Team o solo ad alcuni membri.

Per aumentare la produttività, è stato scelto di attivare l'integrazione di Slack con gli altri tool utilizzati: Asana e GitHub.

I canali scelti di utilizzare sono:

- **docs:** Per condividere e per avere in facile reperibilità le documentazioni di scopo informativo per lo sviluppo del sistema;
- **capitolati:** Per la stesura dello *Studio di fattibilità*;
- **general:** Per discutere gli argomenti di studio individuale;



- **asana**: Per visualizzare i compiti di una attività assegnati ad ogni membro del gruppo;
- **git**: Per le notifiche del Bot sui cambiamenti avvenuti su GitHub e relative domande riguardante ad esso;
- **glossario** Canale per la stesura del *Glossario*;
- **issues**: Per la segnalazione di errori;
- **link**: Per la condivisione di link utili allo studio del progetto;
- **norme**: Canale relativo alle *Norme di progetto*;
- **tex**: Dedicato alla segnalazione di errori riguardanti i file  $\text{\LaTeX}$ .

Un'altra possibilità di comunicazione usata solo per le organizzazioni sui ritrovi interni, è il gruppo creato su Telegram, un servizio di messaggistica istantanea basato su cloud, denominato "SWE - WarMachine", a cui hanno accesso solo i componenti del gruppo. In caso sia necessario effettuare delle videoconferenze, verrà utilizzata l'applicazione Google Hangouts.

#### 4.1.5.2 Comunicazioni esterne

E' compito del *Responsabile di progetto* occuparsi dei contatti con le componenti esterne al gruppo attraverso la casella email di Google creata appositamente per il team: `warmachine.swe@gmail.com`. Se necessario poi, sarà lui stesso ad informare gli altri membri del gruppo su quanto discusso con le componenti esterne.

I contatti delle componenti esterne a cui si può fare riferimento sono:

- **tullio.vardanega@math.unipd.it**: Email del committente;
- **riccardocardin@math.unipd.it**: Email del committente;
- **Gregorio.Piccoli@zucchetti.it**: Email del contatto dell'azienda *Zucchetti s.p.a.*

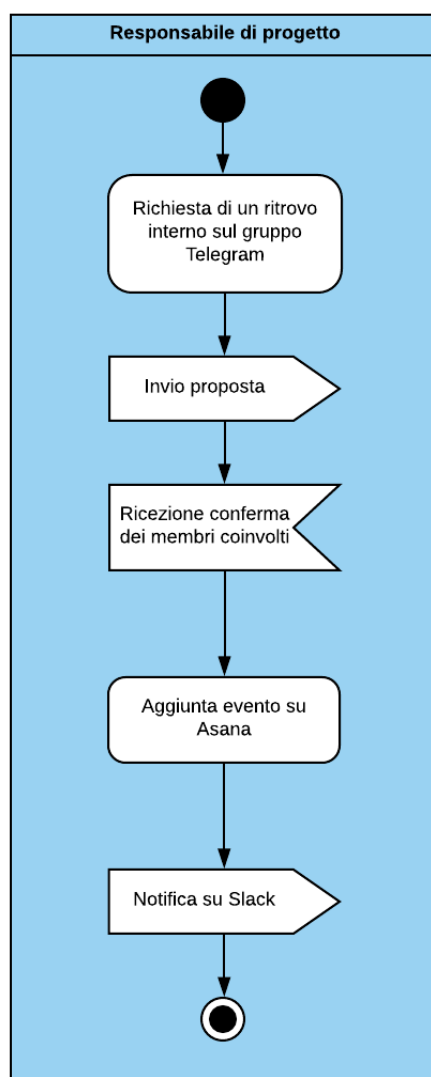
Un altro strumento di comunicazione di cui si prendono impegno tutti i membri, sono i gruppi Telegram:

- **"Coordinamento capitolato 5"**: Per dubbi e organizzazione sui requisiti, incontri con l'azienda *Zucchetti s.p.a.*;
- **"Coordinamento SWE 17/18"**: Per le comunicazioni con tutti i gruppi entranti al secondo lotto sulla scelta dei capitolati e l'organizzazione degli incontri con i committenti.

#### 4.1.6 Gestione degli incontri

##### 4.1.6.1 Incontri interni

Il *Responsabile di progetto* ha il compito di organizzare gli incontri interni, utilizzando Telegram, l'applicazione aderente alle norme (espressa nelle *Comunicazioni interne*, vedi 4.1.5.1), in modo da avere conferma della presenza di ogni membro. Successivamente dovrà fissare l'evento su Asana. Ogni membro del gruppo può richiedere al *Responsabile di progetto* di organizzare un incontro interno segnalando una motivazione e sarà compito di quest'ultimo controllare la disponibilità degli altri membri, e in caso accettare o rifiutare.



**Figura 5:** Procedura per l'organizzazione di un ritrovo interno.

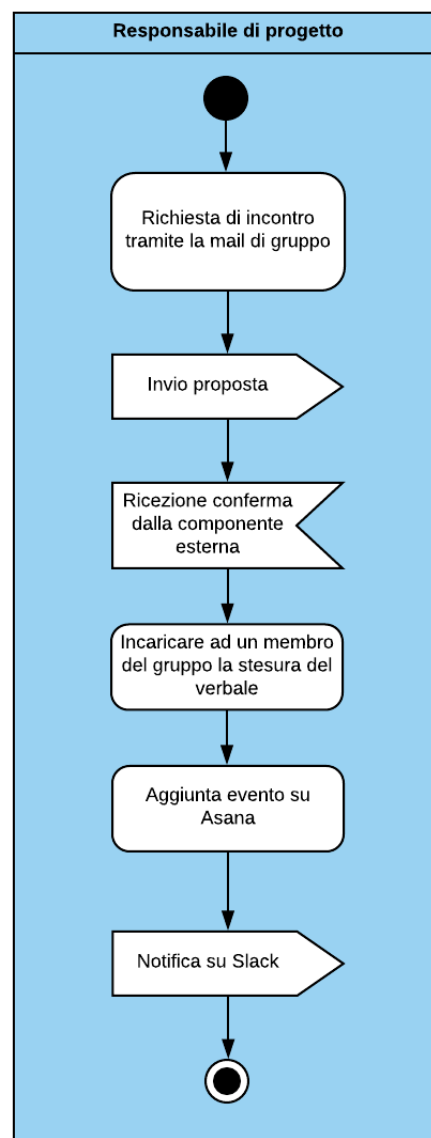


#### 4.1.6.2 Incontri esterni

Il *Responsabile di progetto* seguendo le norme (espresse nelle *Comunicazioni esterne*, vedi 4.1.5.2) deve prendere atto a:

- Fissare l'incontro sul servizio di Asana;
- Incaricare un membro del gruppo per stilare il verbale dell'incontro.

Ogni membro del gruppo può richiedere al *Responsabile di progetto* di organizzare un incontro esterno segnalando una motivazione e sarà compito di quest'ultimo controllare la disponibilità degli altri membri, delle componenti esterne e in caso accettare o rifiutare.



**Figura 6:** Procedura per l'organizzazione di un ritrovo esterno.



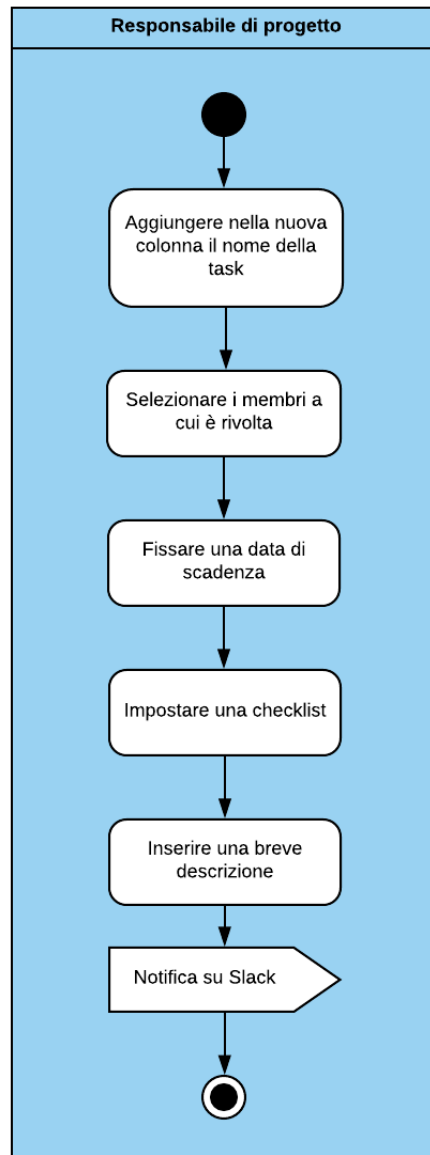
### 4.1.7 Strumenti di coordinamento

#### 4.1.7.1 Ticketing

Per distribuire il carico di lavoro, il *Responsabile di progetto* assegna le attività fra i membri grazie ad Asana, un' applicazione web per la gestione di progetti. Ogni membro dopo aver effettuato la registrazione, avrà accesso alla bacheca del gruppo "WarMachineSwe" contenente un board con una serie di colonne rappresentati le attività da svolgere per la revisione corrente. Per assegnare un ticket il processo è il seguente:

- Cliccare "Aggiungi colonna" e scrivere il nome dell'attività;
- Aggiungere i membri a cui tale attività deve essere assegnata;
- Fissare una data di scadenza concordata con i membri del gruppo;
- Impostare una checklist con l'elenco delle sezioni;
- Cliccare sul simbolo "+" e inserire una descrizione contenente i compiti necessari allo svolgimento dell'attività e il ruolo assunto in quella fase del progetto.

La persona a cui è stata assegnata l'attività riceverà la notifica sul canale Slack dedicato seguendo le norme indicate precedentemente su 4.1.5.1.



**Figura 7:** Procedura per l'assegnazione di un ticket.



#### 4.1.8 Gestione dei rischi

Il *Responsabile di progetto* ha il compito di rilevare i rischi indicati nel documento *Piano di progetto v1.0.0* e nel caso ne vengano individuati di nuovi deve aggiungerli alla sezione di analisi dei rischi. La procedura per la gestione segue il seguente schema:

- Individuare i problemi non calcolati e monitorare i rischi già previsti;
- Registrare ogni riscontro dei rischi nel *Piano di progetto v1.0.0*;
- Aggiungere se sono stati rilevati nuovi rischi nel *Piano di progetto v1.0.0*;
- Ridefinire, se necessario, le strategie di progetto.

#### 4.1.9 Strumenti

##### 4.1.9.1 Sistemi operativi

Il gruppo opera su diversi sistemi operativi:

- MacOS High Sierra 10.13;
- GNU/ Linux<sub>G</sub> almeno 14.04 LTS;
- Windows 10 Pro;
- Windows 10 Home.

##### 4.1.9.2 Slack

Slack è uno strumento di collaborazione utile per le comunicazioni interne ad un gruppo di lavoro. L'applicazione appare come un comune sistema di messaggistica, con in più la possibilità di integrare numerosi servizi tra cui GitHub e Gmail. Gli utenti del team possono suddividere gli argomenti di discussione utilizzando degli hashtag<sub>G</sub> per organizzare al meglio la comunicazione. L'applicazione è gratuita ed è necessaria la registrazione di un dominio per il proprio Team. Una volta registrati gli utenti possono essere invitati all'interno del gruppo. Lo spazio dedicato al gruppo si trova al seguente indirizzo:

<https://swe-group-17.slack.com>

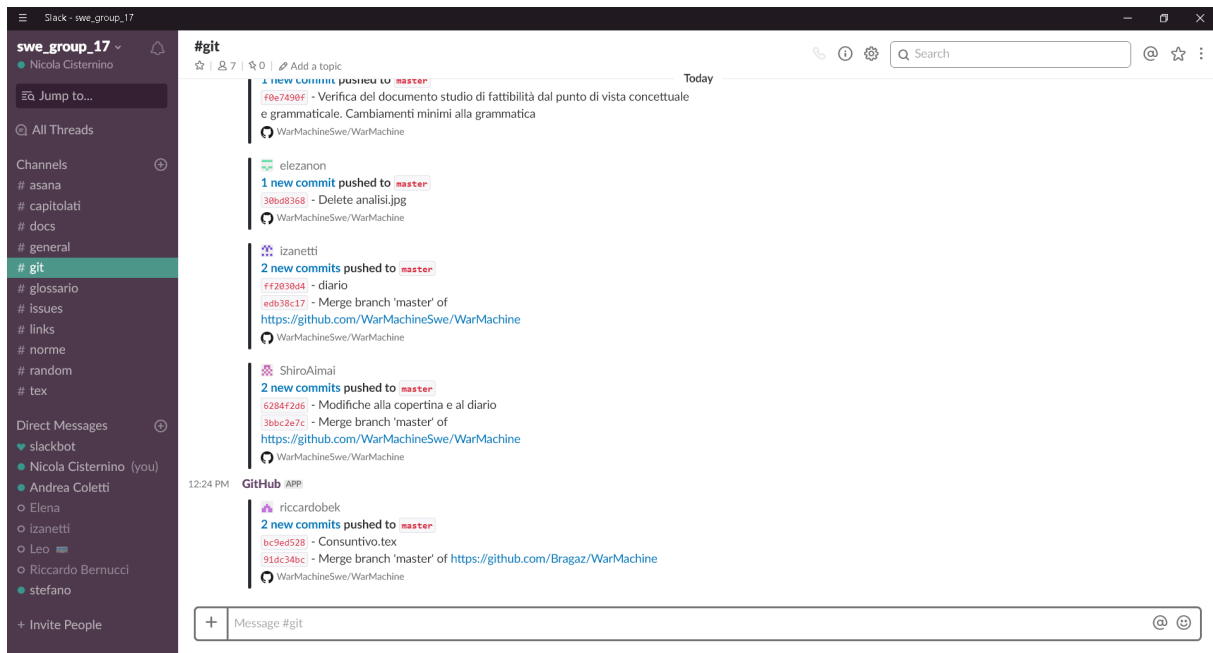


Figura 8: Immagine rappresentativa del profilo Slack del Team WarMachine.

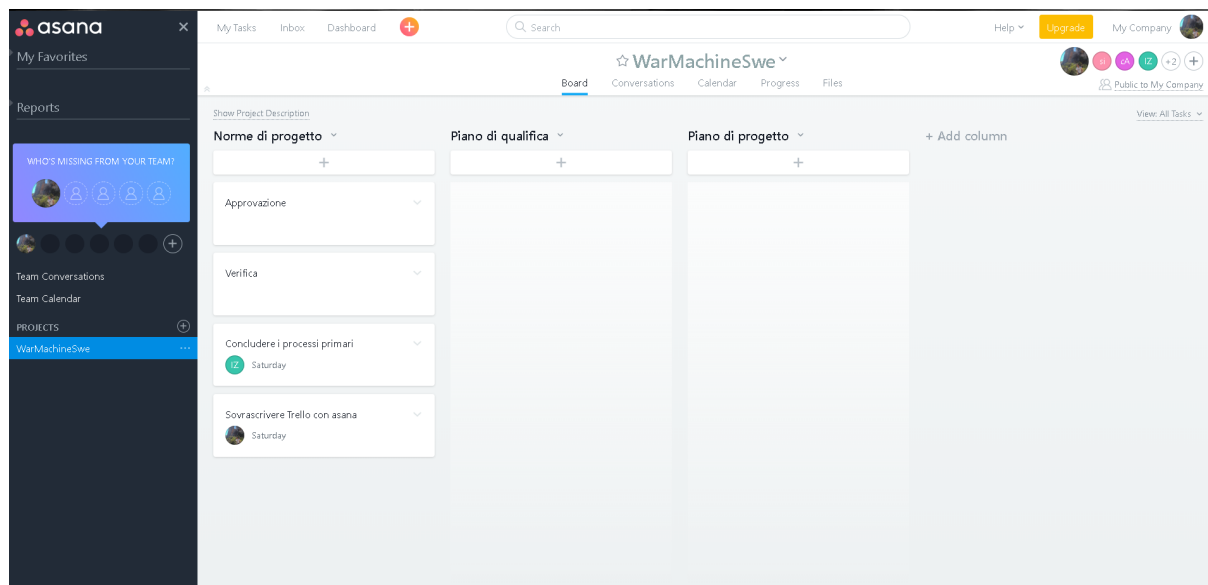
### 4.1.9.3 Telegram

Telegram è un servizio di messaggistica istantanea, usato per le comunicazioni interne ed esterne informali secondo le norme indicate. I client<sub>G</sub> ufficiali di Telegram sono distribuiti come software libero per molteplici piattaforme.

### 4.1.9.4 Asana

Asana è un'applicazione web di gestione di progetti, il cui obiettivo è la corretta suddivisione dei carichi di lavoro tra i membri del gruppo. Essa permette di organizzare il progetto in attività, e le attività in compiti assegnabili ai membri del progetto.

Ai compiti deve essere attribuita una data di calendario che indicherà la scadenza prevista entro cui si vorrebbe concludere il compito. L'applicazione web offre una elevata portabilità ed accessibilità, essendo fruibile direttamente dal web gratuitamente, con un design user-friendly<sub>G</sub>.



**Figura 9:** Asana visualizzato sul web.

#### 4.1.9.5 GitHub

GitHub è un servizio web di code hosting basato su git per progetti software. Gli utenti possono interagire tramite un sistema di issues tracking<sub>G</sub>, pull request<sub>G</sub> e commenti che permette di migliorare il codice del repository (pubblico (disponibile gratuitamente) o privato(disponibile a pagamento), molto utilizzati per lo sviluppo di progetti open-source) risolvendo bug o aggiungendo funzionalità.

#### 4.1.9.6 GitHub Desktop

GitHub Desktop è l'estensione desktop per lavorare sui vari repository a cui si ha accesso nel corrispondente servizio web GitHub. Tra le sue funzioni permette di eseguire:

- La visualizzazione delle modifiche apportate al repository;
- Nei vari branch<sub>G</sub> azioni di commit selezionando da una lista i file a cui apportare modifiche;
- Effettuare il merge delle modifiche locali con quelle dello storico.

E' accessibile il download al seguente link:

<https://desktop.github.com>

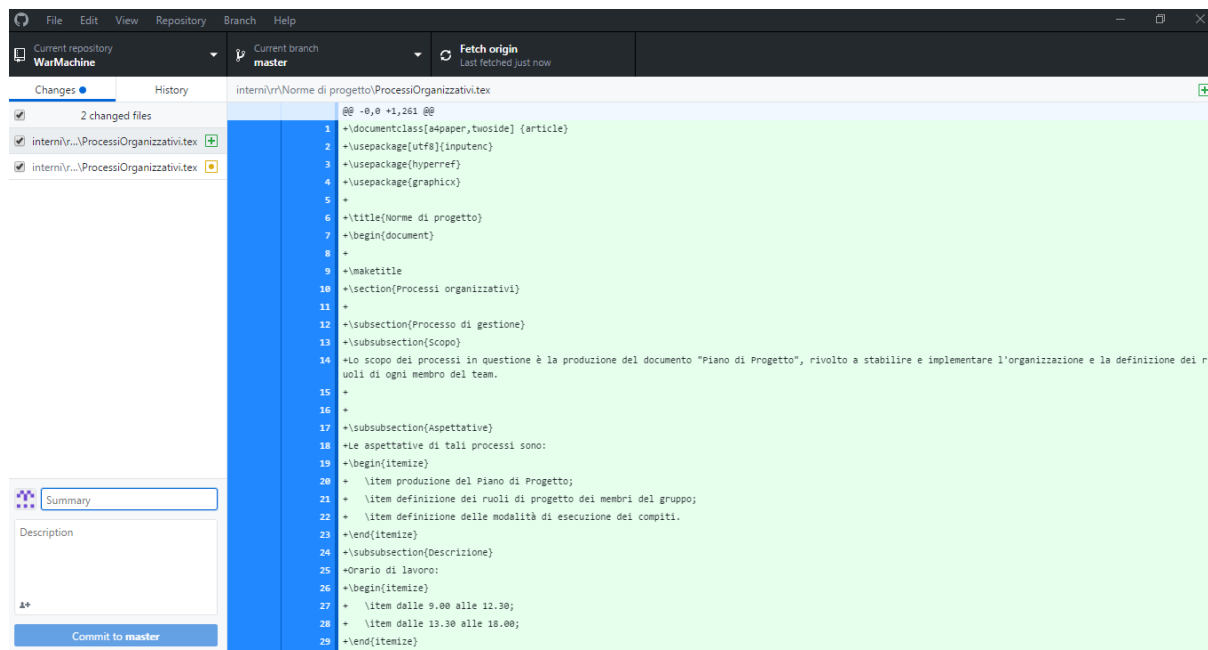


Figura 10: GitHub Desktop su Windows.

### 4.1.9.7 GanttProject

Come strumento per il supporto alla pianificazione di progetto e il corretto allocamento delle risorse è stato scelto GanttProject. GanttProject è uno strumento open-source che permette la creazione diagrammi di Gantt, organizzare i compiti in una work breakdown structure e auto-generare, a partire dai diagrammi di Gantt realizzati, i diagrammi di PERT.

## 4.2 Processo di formazione

Il processo di formazione è responsabilità del singolo individuo. L'individuo deve dedicare parte del suo tempo, che viene concordato internamente al gruppo, all'apprendimento delle tecnologie e delle conoscenze che vengono specificate nel capitolato d'appalto o che il gruppo sceglie di adottare. Il tempo di studio individuale per l'apprendimento viene fissato universalmente all'interno del gruppo, non viene considerato lavoro rendicontato in quanto necessario per la partecipazione al progetto.