

---

# SGAD



*protech.unipd@gmail.com*

## NORME DI PROGETTO v5.00

---

<b>Nome del documento</b>	Norme di Progetto
<b>Versione del documento</b>	5.00
<b>Data redazione</b>	2013/11/28
<b>Redazione</b>	Biancucci Maurizio
<b>Verifica</b>	Gallo Francesco
<b>Approvazione</b>	Segantin Fabio
<b>Uso</b>	Interno
<b>Distribuzione</b>	ProTech

### Sommario

Questo documento vuol definire le norme volte alla regolamentazione delle attività del gruppo ProTech necessarie al processo di sviluppo del progetto SGAD.

## Diario delle modifiche

Modifica	Autore	Ruolo	Data	Versione
<i>Approvazione del documento</i>	Segantin Fabio	Project Manager	2014/03/13	v 5.00
<i>Verifica del documento</i>	Gallo Francesco	Verificatore	2014/03/13	v 4.03
<i>Riorganizzazione del documento</i>	Biancucci Maurizio	Amministratore	2014/03/12	v 4.02
<i>Stesura della sezione "3ds Max 2014"</i>	Biancucci Maurizio	Amministratore	2014/03/12	v 4.01
<i>Approvazione del documento</i>	Nessi Alberto	Project Manager	2014/02/05	v 4.00
<i>Verifica del documento</i>	Battistella Stefano	Verificatore	2014/02/04	v 3.04
<i>Aggiornata sezione verifica</i>	Gallo Francesco	Amministratore	2014/02/29	v 3.03
<i>Aggiornata lista di controllo</i>	Gallo Francesco	Amministratore	2014/02/29	v 3.02
<i>Correzione tempi verbali</i>	Gatto Francesco	Amministratore	2014/02/22	v 3.01
<i>Approvazione del documento</i>	Biancucci Maurizio	Project Manager	2014/01/30	v 3.00
<i>Verifica del documento</i>	Gallo Francesco	Verificatore	2014/01/28	v 2.07
<i>Aggiornata sezione "Lista di controllo", aggiornamento screenshot e descrizione di RACheL</i>	Segantin Fabio	Amministratore	2014/01/25	v 2.06
<i>Aggiornata sezione "Script"</i>	Segantin Fabio	Amministratore	2014/01/22	v 2.05
<i>Riorganizzazione delle sezioni del documento</i>	Segantin Fabio	Amministratore	2014/01/20	v 2.04
<i>Terminata stesura sezione "Tecniche di analisi"</i>	Segantin Fabio	Amministratore	2014/01/16	v 2.03
<i>Ristrutturate le varie sezioni</i>	Segantin Fabio	Amministratore	2014/01/15	v 2.02

<i>Correzioni sull'uso del termine "fase"</i>	Segantin Fabio	Amministratore	2014/01/12	v 2.01
<i>Approvazione del documento</i>	Gallo Francesco	Project Manager	2014/01/06	v 2.00
<i>Verifica del documento</i>	Battistella Stefano	Verificatore	2014/01/05	v 1.02
<i>Passaggio a nuova versione di Visual Paradigm</i>	Biancucci Maurizio	Amministratore	2014/01/30	v 1.01
<i>Approvazione del documento</i>	Segantin Fabio	Project Manager	2013/12/03	v 1.00
<i>Verifica del documento</i>	Gallo Francesco	Verificatore	2013/12/03	v 0.09
<i>Correzione degli errori sezioni "Repository", "Analisi dei requisiti" e "Ambiente di sviluppo"</i>	Biancucci Maurizio	Amministratore	2013/12/03	v 0.08
<i>Correzione degli errori sezioni "Documentazione" e "Organizzazione"</i>	Battistella Stefano	Amministratore	2013/12/03	v 0.07
<i>Verifica del documento</i>	Gallo Francesco	Verificatore	2013/12/02	v 0.06
<i>Terminata stesura sezioni "Analisi dei requisiti" e "Ambiente di sviluppo"</i>	Biancucci Maurizio	Amministratore	2013/12/01	v 0.05
<i>Terminata stesura sezione "Organizzazione"</i>	Battistella Stefano	Amministratore	2013/12/01	v 0.04
<i>Terminata stesura sezione "Repository"</i>	Biancucci Maurizio	Amministratore	2013/11/30	v 0.03
<i>Terminata stesura sezione "Documentazione"</i>	Battistella Stefano	Amministratore	2013/11/29	v 0.02
<i>Inizio stesura documento con impostazione generale dello scheletro del documento</i>	Battistella Stefano	Amministratore	2013/11/28	v 0.01

## Indice

<b>1</b>	<b>Introduzione</b>	<b>7</b>
1.1	Scopo del documento . . . . .	7
1.2	Scopo del prodotto . . . . .	7
1.3	Glossario . . . . .	7
1.4	Riferimenti . . . . .	7
1.4.1	Informativi . . . . .	7
<b>2</b>	<b>Processi primari</b>	<b>9</b>
2.1	Processo di sviluppo . . . . .	9
2.1.1	Attività . . . . .	9
2.1.1.1	Analisi dei Requisiti . . . . .	9
2.1.1.1.1	Task - Studio di Fattibilità . . . . .	9
2.1.1.1.2	Task - Analisi dei Requisiti . . . . .	9
2.1.1.2	Progettazione . . . . .	10
2.1.1.2.1	Task - Specifica Tecnica . . . . .	10
2.1.1.2.2	Task - Definizione di Prodotto . . . . .	10
2.1.1.3	Codifica . . . . .	11
2.1.2	Norme . . . . .	11
2.1.2.1	Classificazione dei requisiti . . . . .	11
2.1.2.2	Classificazione dei casi d'uso . . . . .	12
2.1.2.3	Codifica e convenzioni . . . . .	12
2.1.2.4	Nomi e norme stilistiche . . . . .	12
2.1.2.5	Ricorsione . . . . .	13
2.1.3	Strumenti . . . . .	13
2.1.3.1	RACheL . . . . .	13
2.1.3.1.1	Casi d'uso . . . . .	13
2.1.3.1.2	Requisiti . . . . .	14
2.1.3.1.3	Attori . . . . .	14
2.1.3.1.4	Design pattern . . . . .	14
2.1.3.1.5	Componenti . . . . .	14
2.1.3.1.6	Classi . . . . .	14
2.1.3.1.7	Test . . . . .	14
2.1.3.1.8	Tracciamento . . . . .	14
2.1.3.2	3ds Max 2014 . . . . .	15
2.1.3.3	IntelliJDEA . . . . .	15
2.1.3.4	Eclipse . . . . .	15
2.1.3.5	WebStorm . . . . .	15
<b>3</b>	<b>Processi di supporto</b>	<b>16</b>
3.1	Processo di documentazione . . . . .	16
3.1.1	Procedure . . . . .	16
3.1.1.1	Procedura di formalizzazione dei documenti . . . . .	16
3.1.2	Norme . . . . .	16
3.1.2.1	Template . . . . .	16

3.1.2.2	Norme tipografiche . . . . .	16
3.1.2.2.1	Stile del testo . . . . .	18
3.1.2.2.2	Punteggiatura . . . . .	18
3.1.2.2.3	Composizione del testo . . . . .	19
3.1.2.2.4	Formati ricorrenti . . . . .	20
3.1.2.3	Componenti grafiche . . . . .	21
3.1.2.3.1	Immagini . . . . .	22
3.1.2.3.2	Tabelle . . . . .	22
3.1.2.4	Formattazione dei documenti . . . . .	22
3.1.2.4.1	Frontespizio . . . . .	22
3.1.2.4.2	Diario delle modifiche . . . . .	23
3.1.2.4.3	Indici . . . . .	23
3.1.2.4.4	Intero documento . . . . .	24
3.1.2.5	Tipi di documenti . . . . .	24
3.1.2.5.1	Verbali . . . . .	24
3.1.2.5.2	Documenti informali . . . . .	25
3.1.2.5.3	Documenti formali . . . . .	25
3.1.2.5.4	Glossario . . . . .	25
3.1.2.6	Intestazione file di documentazione . . . . .	25
3.1.2.7	Versionamento dei documenti . . . . .	26
3.1.2.7.1	Avanzamento di versione documenti . . . . .	26
3.1.3	Strumenti . . . . .	26
3.1.3.1	LaTeX . . . . .	26
3.1.3.2	Visual Paradigm . . . . .	27
3.1.3.3	Script . . . . .	27
3.2	Processo di verifica . . . . .	28
3.2.1	Attività . . . . .	28
3.2.1.1	Analisi . . . . .	28
3.2.1.1.1	Analisi statica . . . . .	28
3.2.1.1.2	Analisi dinamica . . . . .	29
3.2.1.2	Test . . . . .	29
3.2.1.2.1	Test di unità . . . . .	29
3.2.1.2.2	Test di integrazione . . . . .	29
3.2.1.2.3	Test di sistema . . . . .	30
3.2.1.2.4	Test di regressione . . . . .	30
3.2.1.2.5	Test di validazione . . . . .	30
3.2.1.3	Tracciamento . . . . .	30
3.2.2	Procedure . . . . .	30
3.2.2.1	Procedura per la gestione delle anomalie . . . . .	30
3.2.3	Strumenti . . . . .	31
3.2.3.1	Correzione ortografica automatica . . . . .	31
3.2.3.2	Hunspell . . . . .	31
3.2.3.3	Calcolo Gulpease . . . . .	33
3.2.3.4	Strumenti per la validazione W3C . . . . .	33
3.2.3.5	Tracker . . . . .	33
3.2.3.5.1	Metriche . . . . .	33

<b>4</b>	<b>Processi organizzativi</b>	<b>35</b>
4.1	Processo di gestione	35
4.1.1	Attività	35
4.1.1.1	Comunicazioni	35
4.1.1.1.1	Comunicazioni interne	35
4.1.1.1.2	Comunicazioni esterne	35
4.1.1.2	Gestione incontri	35
4.1.1.2.1	Incontri interni	35
4.1.1.2.2	Incontri esterni	36
4.1.1.3	Ticketing	36
4.1.1.4	Repository	36
4.1.2	Procedure	36
4.1.2.1	Procedura di assegnazione	36
4.1.2.2	Procedura per la rilevazione dei rischi	38
4.1.3	Norme	38
4.1.3.1	Regole generali	38
4.1.3.2	Protocollo di utilizzo	39
4.1.3.3	Ruoli di progetto	39
4.1.3.3.1	Project Manager	41
4.1.3.3.2	Amministratore	41
4.1.3.3.3	Analista	42
4.1.3.3.4	Progettista	42
4.1.3.3.5	Programmatore	42
4.1.3.3.6	Verificatore	43
4.1.3.4	Repository	43
4.1.3.4.1	Norme sui nomi dei file	43
4.1.3.4.2	Struttura del repository	43
4.1.3.4.3	Norme sulla commit	44
4.1.4	Strumenti	44
4.1.4.1	Sistema operativo	44
4.1.4.2	Dropbox	44
4.1.4.3	Google Calendar	44
4.1.4.4	MyTinyToDo	45
4.1.4.5	Facebook	45
4.1.4.6	ProjectLibre	45
4.1.4.7	GitHub	45
4.1.4.8	Git	45
4.1.4.9	Prezi	45
<b>A</b>	<b>Lista di controllo</b>	<b>46</b>
<b>B</b>	<b>Screenshot MyTinyToDo</b>	<b>49</b>
<b>C</b>	<b>Screenshot RACheL</b>	<b>51</b>
<b>D</b>	<b>Screenshot Tracker</b>	<b>62</b>

## Elenco delle figure

1	Diagramma di attività - formalizzazione dei documenti . . . . .	17
2	Diagramma di attività - procedura per la gestione delle anomalie . . . . .	32
3	Diagramma di attività - assegnazione di un ticket . . . . .	37
4	Diagramma di attività - svolgimento di un ticket . . . . .	40
5	Vista generale della pagina iniziale di MyTinyToDo . . . . .	49
6	Funzione di filtro ticket per tag . . . . .	49
7	Form di creazione di un ticket . . . . .	50
8	Analisi della sintassi di un ticket . . . . .	50
9	Pagina principale di RACheL con segnalazione errori sui dati . . . . .	51
10	Pagina alert di RACheL . . . . .	52
11	Pagina di ricapitolazione Use Case . . . . .	52
12	Pagina per Use Case . . . . .	53
13	Pagina di inserimento Use Case . . . . .	54
14	Pagina ricapitolazione requisiti . . . . .	55
15	Pagina relativa ad un requisito . . . . .	56
16	Pagina di inserimento requisiti . . . . .	57
17	Pagina di riepilogo attori . . . . .	57
18	Pagina relativa ad un attore . . . . .	58
19	Pagina relativa ad un design pattern . . . . .	59
20	Pagina relativa ad un componente . . . . .	60
21	Pagina relativa ad una classe . . . . .	61
22	Pagina principale di Tracker . . . . .	62
23	Pagina relativa agli errori di un documento . . . . .	63

## 1 Introduzione

### 1.1 Scopo del documento

Questo documento si prefigge di definire le varie norme che regolamentano l'intero svolgimento del progetto. Tutti i membri del *team<sub>|g|</sub>* sono obbligati a visionare tale documento e a sottostare alle norme ivi descritte. Tali norme permettono di migliorare: la coerenza e la consistenza dei file prodotti, l'efficienza del lavoro e dell'esecuzione delle operazioni di verifica.

In particolare si tratteranno:

- le interazioni tra i membri del team;
- le interazioni con componenti esterne al team;
- la definizione dei ruoli e l'identificazione delle mansioni relative;
- le convenzioni tipografiche e le modalità di stesura delle documentazioni;
- le modalità di lavoro durante le varie fasi del progetto;
- gli ambienti di sviluppo, il *repository<sub>|g|</sub>* e il *ticketing<sub>|g|</sub>*.

Nel caso in cui ve ne sia necessità, ogni membro del team potrà contattare il *Project Manager<sub>|g|</sub>* per suggerire eventuali cambiamenti.

### 1.2 Scopo del prodotto

Lo scopo del *prodotto<sub>|g|</sub>* è di creare un'architettura distribuita. Tale architettura dovrà servire per la gestione efficiente delle operazioni generate da utenti dei *social game<sub>|g|</sub>*. Il prodotto comprenderà un piccolo *browser game<sub>|g|</sub>*. Esso sarà di tipo gestionale ed utilizzerà l'architettura server prodotta.

### 1.3 Glossario

Per evitare ogni ambiguità relativa al linguaggio e ai termini utilizzati nei documenti formali, si allega il "*Glossario v2.00*". In questo documento vengono definiti e descritti tutti i termini con un significato particolare. Per rendere più facile la lettura, i termini saranno posti in corsivo e accanto a questi ci sarà una 'g' corsiva, compresa tra barre verticali, a pedice (esempio: *Glossario<sub>|g|</sub>*).

### 1.4 Riferimenti

#### 1.4.1 Informativi

- **Analisi dei Requisiti:** "*Analisi dei Requisiti v6.00*";
- **Piano di Progetto:** "*Piano di Progetto v5.00*";
- **Piano di Qualifica:** "*Piano di Qualifica v5.00*";
- **Studio di Fattibilità:** "*Studio di Fattibilità v1.00*";



- [SI/ISO 31-0]: [http://en.wikipedia.org/wiki/ISO\\_31-0](http://en.wikipedia.org/wiki/ISO_31-0);
- [ISO 8601]: [http://en.wikipedia.org/wiki/ISO\\_8601](http://en.wikipedia.org/wiki/ISO_8601);
- GitHub: <http://github.com/>;
- ANSI: <http://www.ansi.org/>.

## 2 Processi primari

### 2.1 Processo di sviluppo

#### 2.1.1 Attività

##### 2.1.1.1 Analisi dei Requisiti

###### 2.1.1.1.1 Task - Studio di Fattibilità

È compito del Project Manager organizzare riunioni per permettere il confronto e lo scambio di opinioni tra i membri del team sui capitolati. Il primo documento per l'analisi dei requisiti è lo “*Studio di Fattibilità v1.00*” la cui realizzazione è affidata agli Analisti. Essi si baseranno su quanto deciso nelle riunioni e scriveranno un'analisi che descriva i seguenti punti:

- **dominio tecnologico e applicativo:** si pensa a quanto sono conosciute le tecnologie richieste. Nella valutazione si considerano quali sono le attuali conoscenze del team.
- **interesse strategico:** si valuta l'interesse strategico del team; esso viene valutato in relazione al *capitolato*<sub>|g|</sub> in analisi.
- **individuazione dei rischi:** si comprendono quali sono i punti critici della realizzazione. Si individuano eventuali rischi.

###### 2.1.1.1.2 Task - Analisi dei Requisiti

Dopo aver concluso lo studio di fattibilità, sarà compito degli Analisti scrivere il documento “*Analisi dei Requisiti v6.00*”.

Lo scopo principale di questa attività è quella di produrre dei requisiti semplici e di facile comprensione, a partire da tutte le informazioni recuperabili.

Per automatizzare e velocizzare il più possibile questa attività, è stato creato dal team il software *RACheL*<sub>|g|</sub><sup>1</sup>, in cui andranno inseriti tutti i requisiti.

Le risorse che si possono analizzare per ottenere informazioni sui requisiti, ordinate per importanza in ordine decrescente, sono le seguenti:

1. il capitolato d'appalto;
2. incontri con il *proponente*<sub>|g|</sub>;
3. incontri con il *committente*<sub>|g|</sub>.

Ci si aspetta inoltre che vengano definiti i vari test da dover eseguire per provare che il *sistema*<sub>|g|</sub> rispetti i requisiti specificati.

- **Input:** il capitolato d'appalto e le informazioni recuperate dagli *stakeholder*<sub>|g|</sub>;
- **Output:** documento “*Analisi dei Requisiti v6.00*”, test di sistema;
- **Risorse:** analisi, strumentazione.

---

<sup>1</sup>Per ulteriori spiegazioni su RACheL si rimanda alla sezione 2.1.3.1 RACheL.

### 2.1.1.2 Progettazione

#### 2.1.1.2.1 Task - Specifica Tecnica

I Progettisti devono descrivere la progettazione ad alto livello dell'architettura dell'applicazione e dei singoli componenti nella “*Specifica Tecnica v3.00*”. Devono inoltre provvedere alla progettazione di opportuni test di integrazione.

- Diagrammi  $UML_{|g|}$

Devono essere realizzati i seguenti diagrammi:

- diagrammi delle classi;
- diagrammi dei  $package_{|g|}$ ;
- diagrammi di attività;
- diagrammi di sequenza.

- $Design\ pattern_{|g|}$

I Progettisti devono descrivere i design pattern utilizzati per realizzare l'architettura. Di tali design pattern, si deve includere una breve descrizione e un diagramma che ne esemplifichi il funzionamento e la struttura.

- Tracciamento componenti

Ogni requisito deve essere tracciato al componente che lo soddisfa. L'applicazione web RACHel genera automaticamente le tabelle di tracciamento, come descritto nella sottosezione 3.2.1.3.

In questo modo sarà possibile garantire che ogni requisito venga soddisfatto e, al tempo stesso, misurare il progresso nell'attività di progettazione.

- Test di integrazione

I Progettisti devono definire delle classi di verifica. Tali classi sono necessarie per verificare che i componenti del sistema funzionino nella maniera prevista.

#### 2.1.1.2.2 Task - Definizione di Prodotto

I Progettisti devono produrre la “*Definizione di Prodotto*”. In essa viene descritta la progettazione di dettaglio del sistema, ampliando quanto scritto nella “*Specifica Tecnica v3.00*”.

- Diagrammi UML

Devono essere redatti i seguenti diagrammi:

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

- Definizione di classe

Ogni classe progettata deve essere descritta all'interno della “*Definizione di Prodotto*”. Tale descrizione deve comprendere una spiegazione sullo scopo della classe e deve specificare quale funzionalità essa modella. Nella descrizione devono inoltre essere presenti l'elenco di metodi e attributi della classe.

- **Tracciamento classi**

Ogni requisito deve essere tracciato alle classi che lo soddisfano. Il software RACheL genera in automatico le tabelle di tracciamento, come descritto nella sotto-sezione 3.2.1.3. In questo modo sarà possibile misurare il progresso nell'attività di progettazione e garantire che ogni classe soddisfi almeno un requisito.

- **Test di unità**

I Progettisti devono definire i test di unità necessari per verificare che i componenti del sistema funzionino nel modo previsto.

### 2.1.1.3 Codifica

La progettazione del prodotto sarà seguita dall'attività di codifica.

## 2.1.2 Norme

### 2.1.2.1 Classificazione dei requisiti

I requisiti prodotti devono essere classificati per tipo e per importanza secondo la seguente codifica:

$$R[\text{Importanza}][\text{Tipo}][\text{Codice}]$$

- **Importanza:** può assumere solo uno fra i seguenti valori:

- 0: requisito obbligatorio;
- 1: requisito desiderabile;
- 2: requisito opzionale.

- **Tipo:** può assumere solo uno fra i seguenti valori:

- *F*: funzionale;
- *Q*: di qualità;
- *P*: prestazionale;
- *V*: vincolo.

- **Codice:** è il codice gerarchico univoco di ogni vincolo espresso in numeri (esempio: 1.3.2).

Per ogni requisito vengono inoltre specificati:

- **descrizione:** breve ma completa ed il meno ambigua possibile;

- **fonte:** può essere soltanto una o più tra le seguenti:

- *capitolato*: si tratta di un requisito direttamente estrapolato dal capitolato.
- *interno*: gli Analisti hanno ritenuto giusto aggiungere questo requisito.
- *caso d'uso*: il requisito è stato estrapolato da un caso d'uso. In questo caso va indicato il codice univoco del caso d'uso. È possibile indicare come fonte più di un caso d'uso.

### 2.1.2.2 Classificazione dei casi d'uso

I casi d'uso prodotti devono essere suddivisi in ordine gerarchico usando la seguente codifica:

UC[Codice]

- **Codice:** è il codice gerarchico univoco di ogni vincolo espresso in numeri (esempio: 1.3.2).

Per ogni caso d'uso vengono inoltre specificati:

- **attori coinvolti:** lista degli attori coinvolti nel caso d'uso;
- **descrizione:** scopo e descrizione del caso d'uso, breve ma completa ed il meno ambigua possibile;
- **precondizione:** condizione di partenza sempre vera dalla quale il caso d'uso parte;
- **flusso principale degli eventi:** sezione in cui si descrive il flusso dei casi d'uso figli;
- **estensioni:** spiegazione di tutte le estensioni se presenti;
- **inclusioni:** spiegazione di tutte le inclusioni se presenti;
- **generalizzazioni:** spiegazione di tutte le generalizzazioni, se presenti;
- **postcondizione:** la condizione finale sempre vera alla fine dell'esecuzione del caso d'uso.

### 2.1.2.3 Codifica e convenzioni

- Tutti i file contenenti codice o documentazione devono essere conformi alla codifica  $UTF_{|g|}-8$  senza  $BOM_{|g|}$  (poiché potrebbe causare problemi nella verifica). Per andare a capo viene usato il carattere LF (U+000A).
- È ammessa la possibilità di effettuare modifiche alle convenzioni stabilite, in seguito ad una decisione del Project Manager.

### 2.1.2.4 Nomi e norme stilistiche

- Gli sviluppatori devono rispettare le regole di indentazione disponibili all'indirizzo [http://en.wikipedia.org/wiki/Programming\\_style](http://en.wikipedia.org/wiki/Programming_style);
- I nomi di variabili, classi, funzioni e metodi devono essere in inglese e privi di underscore;
- I commenti devono essere scritti in lingua italiana;
- I nomi delle classi devono avere la prima lettera maiuscola;
- I nomi di variabili, metodi e funzioni devono avere la prima lettera minuscola e le successive iniziali delle parole che ne compongono il nome, in maiuscolo.

### 2.1.2.5 Ricorsione

La ricorsione va evitata quando possibile. Per ogni funzione ricorsiva è necessario fornire una prova di terminazione. È inoltre necessario valutare il costo in termini di occupazione della memoria. Nel caso in cui l'utilizzo di memoria risulti troppo elevato, la ricorsione verrà rimossa.

### 2.1.3 Strumenti

#### 2.1.3.1 RACheL

Al fine di gestire in maniera veloce e il più possibile automatizzata tutti i dati, e per semplificare i tracciamenti, il team ha creato un applicativo web: RACheL.

RACheL è accessibile tramite *browser<sub>|g|</sub>*, ed è ospitato nel server virtuale del team insieme ad altri strumenti, indicati nelle sottosezioni 4.1.4.4 e 3.2.3.5. Ogni membro del team può accedervi previa autenticazione (per comodità è stato creato un unico account comune). Tutti i dati interfacciati sono presi direttamente da un *database<sub>|g|</sub> MySQL<sub>|g|</sub>*. È possibile inserire, modificare e eliminare ogni dato direttamente dall'applicativo.

Grazie ad opportuni controlli, la gestione dei dati è costantemente monitorata. L'inserimento e la modifica sono soggetti a verifica mediante *trigger<sub>|g|</sub>*, che agevolano la consistenza e la coerenza delle informazioni. Inoltre, ogniqualvolta si presenti una difformità<sup>2</sup>, nella home page dell'applicativo appare un link di avviso che notifica la presenza di un problema<sup>3</sup>. Questo permette di accedere ad una apposita pagina che mostra tutti i problemi rilevati<sup>4</sup>.

Tutti i requisiti e i casi d'uso sono stati prima organizzati nel database e successivamente interfacciati in RACheL. Dalla home page, nella tabella di sinistra, è possibile scegliere in quale sezione entrare tra Casi d'Uso, Requisiti e Attori<sup>10</sup>. Inoltre, è stata automatizzata la stesura delle parti dell'“*Analisi dei Requisiti v6.00*”, rendendo disponibile l'esportazione del codice *L<sup>A</sup>T<sub>E</sub>X<sub>|g|</sub>*. RACheL consente la visualizzazione, la modifica e l'eliminazione di design pattern, componenti, classi e test studiati dai Progettisti. Di questi, ne automatizza i tracciamenti, come evidenziato nella sottosezione 3.2.1.3. Per ognuna di queste categorie è stata creata una pagina, raggiungibile dalla rispettiva voce nel menù di destra della home page. In ogni pagina è presente un bottone per l'inserimento di un nuovo record nel database.

RACheL inoltre calcola le dipendenze tra le classi, e le dipendenze fra le componenti. Infine, tiene traccia della struttura gerarchica tra ogni componente e le relative classi.

#### 2.1.3.1.1 Casi d'uso

La pagina dei casi d'uso traccia tutti i casi d'uso rilevati dagli Analisti<sup>11</sup>. Questi sono opportunamente elencati in ordine gerarchico, con rispettivo titolo e tipologia. Ogni codice identificativo, inoltre, è un link alla rispettiva pagina del caso d'uso<sup>5</sup>. Qui sono raggruppate tutte le informazioni e le operazioni disponibili. Infine, è possibile aggiungere un nuovo caso d'uso semplicemente cliccando sul bottone “Inserisci UC”. Questo rimanda ad una apposita pagina contenente un *form<sub>|g|</sub>* per l'inserimento dei dati<sup>6</sup>.

---

<sup>2</sup>Ad esempio, dei campi dati lasciati vuoti, dei requisiti non mappati, etc.

<sup>3</sup>Si veda la figura 9 in Appendice

<sup>4</sup>Si veda la figura 10 in Appendice

<sup>5</sup>Si veda la figura 11 in Appendice.

<sup>6</sup>Si veda la figura 12 in Appendice.

#### 2.1.3.1.2 Requisiti

La seconda voce del menù rimanda alla pagina dei requisiti<sup>7</sup>. Qui sono elencati tutti i requisiti suddivisi per tipologia (RF: Requisiti Funzionali, RP: Requisiti Prestazionali, RQ: Requisiti di Qualità, RV: Requisiti di Verifica). Come per i casi d'uso, anche i requisiti hanno un codice identificativo che rimanda alla rispettiva pagina informativa<sup>8</sup>. Con il bottone "Insert requirement" si accede alla pagina di inserimento di un nuovo requisito<sup>9</sup>.

#### 2.1.3.1.3 Attori

L'ultima voce del menù di sinistra, infine, riferisce gli attori del sistema. Il *layout*<sub>|g|</sub> è uguale a quello di casi d'uso e requisiti, con funzionalità analoghe<sup>10</sup>.

#### 2.1.3.1.4 Design pattern

Nella pagina dei design pattern, questi sono suddivisi per tipologia<sup>11</sup>. Per ognuna è stata creata una tabella che visualizza l'identificativo del design pattern, la sua descrizione e un bottone per l'eliminazione.

#### 2.1.3.1.5 Componenti

I componenti sono elencati in un'unica tabella, in ordine gerarchico col rispettivo identificativo. Anche per essi è stato messo a disposizione un bottone di eliminazione del componente.

#### 2.1.3.1.6 Classi

Le classi sono elencate in modo analogo ai componenti, rispettando lo stesso layout.

#### 2.1.3.1.7 Test

Con RACheL infine è possibile tracciare tutti i test di integrazione, di sistema e di validazione. L'apposita pagina elenca tutti i test inseriti fino a quel momento e la relativa descrizione. Per ognuno di essi, è disponibile un bottone per l'eliminazione dal database.

#### 2.1.3.1.8 Tracciamento

RACheL consente di esportare direttamente in  $\text{\LaTeX}$  molte parti dei documenti del progetto, automatizzando il processo di stesura di questi. Grazie ad uno *script*<sub>|g|</sub> appositamente creato, è possibile in ogni momento scaricare tutti i file  $\text{\LaTeX}$ , inserendoli nelle cartelle dei rispettivi documenti.

RACheL consente il tracciamento delle delle seguenti coppie di elementi:

- Requisiti - Fonti;
- Fonti - Requisiti;
- Requisiti - Componenti;
- Componenti - Requisiti;

---

<sup>7</sup>Si veda la figura 13 in Appendice.

<sup>8</sup>Si veda la figura 14 in Appendice.

<sup>9</sup>Si veda la figura 15 in Appendice.

<sup>10</sup>Si veda la figura 16 e figura 17 in Appendice.

<sup>11</sup>Sono stati utilizzati design pattern architetturali, comportamentali, creazionali e strutturali.

- Componenti - Classi;
- Classi - Componenti;
- Requisiti - Test di Sistema;
- Test di Sistema - Requisiti;
- Test di Validazione - Requisiti;
- Test di Integrazione - Componenti;
- Componenti - Test di Integrazione;
- Design Pattern - Classi;
- Classi - Design Pattern.

#### **2.1.3.2 3ds Max 2014**

Il software 3ds Max 2014, con licenza student, è stato utilizzato dal gruppo per creare i modelli degli edifici rappresentati nel mondo di gioco.

#### **2.1.3.3 IntelliJDEA**

L'IDE<sub>|g|</sub> IntelliJDEA 13.0 Community Edition è utilizzato per scrivere il codice nel linguaggio di programmazione *Scala*<sub>|g|</sub>.

#### **2.1.3.4 Eclipse**

L' IDE Eclipse 4.3 Kepler è utilizzato per scrivere codice nel linguaggio di programmazione *Java*<sub>|g|</sub>.

#### **2.1.3.5 WebStorm**

L'IDE JetBrains WebStorm 7.0.3 è utilizzato per scrivere codice nel linguaggio di programmazione *JavaScript*<sub>|g|</sub>.



## 3 Processi di supporto

### 3.1 Processo di documentazione

In questa sezione verranno presentati gli standard che riguardano i documenti prodotti durante il ciclo di vita del prodotto. Tali documenti sono stati prodotti dal gruppo ProTech.

#### 3.1.1 Procedure

##### 3.1.1.1 Procedura di formalizzazione dei documenti

Quando si ritiene conclusa la stesura di un documento non formale, viene chiamato in causa il Project Manager. È sua responsabilità assegnare il documento ai Verificatori.

Nel caso in cui i Verificatori individuino degli errori o delle difformità nel documento, sarà loro dovere informare il Project Manager che rimetterà il documento nelle mani dei redattori. Il ciclo si ripete finché i Verificatori non hanno più nulla da segnalare.

Il documento verrà quindi passato al Project Manager, che sarà responsabile dell'approvazione o meno del documento. Nel caso decidesse di respingerlo, comunicherà ai redattori le modifiche da apportare. In casi estremi, potrà comunicare che la stesura del documento dovrà essere fatta ex-novo.

Se il documento verrà approvato lo si riterrà un documento formale, e potrà essere distribuito alle persone nominate nella lista di distribuzione.

La procedura di formalizzazione viene riassunta nel diagramma delle attività indicato come da figura 1.

#### 3.1.2 Norme

##### 3.1.2.1 Template

Ogni documento dovrà essere realizzato a partire da un *template*<sub>|g|</sub> L<sup>A</sup>T<sub>E</sub>X appropriato reperibile al percorso `$/aSgad/Template_Latex/Template/$`. In base al tipo di documento si dovrà scegliere il template corretto, essendo l'elenco dei comandi personalizzati diverso per i vari documenti.

Lo scopo dei template è quello di permettere, a colui che redige il documento, di adottare automaticamente le conformità previste dalle “*Norme di Progetto v5.00*”. Permette inoltre di agevolare la procedura di adeguamento alle nuove norme per la redazione, nel caso esse cambino. Infatti, è possibile modificare solamente i file che contengono i vari comandi creati appositamente e rendere uniformi tutti i documenti che ne fanno uso.

Se i membri del team vogliono segnalare errori o modifiche, si rimanda all'Amministratore. Tali segnalazioni possono riguardare i template ed i comandi personalizzati. Egli provvederà ad informare tutti i componenti del team in merito alle modifiche apportate. Si richiede l'impegno da parte di ogni membro di riferirsi al documento “*Guida Comandi L<sup>A</sup>T<sub>E</sub>X v1.00*” per apprendere la funzione e l'uso dei vari comandi personalizzati in L<sup>A</sup>T<sub>E</sub>X.

##### 3.1.2.2 Norme tipografiche

Al fine di evitare incongruenze tra i vari file, si rimanda a questa sezione. Le norme riguardano l'ortografia, la tipografia e l'assunzione di uno stile uniforme in tutti i documenti.

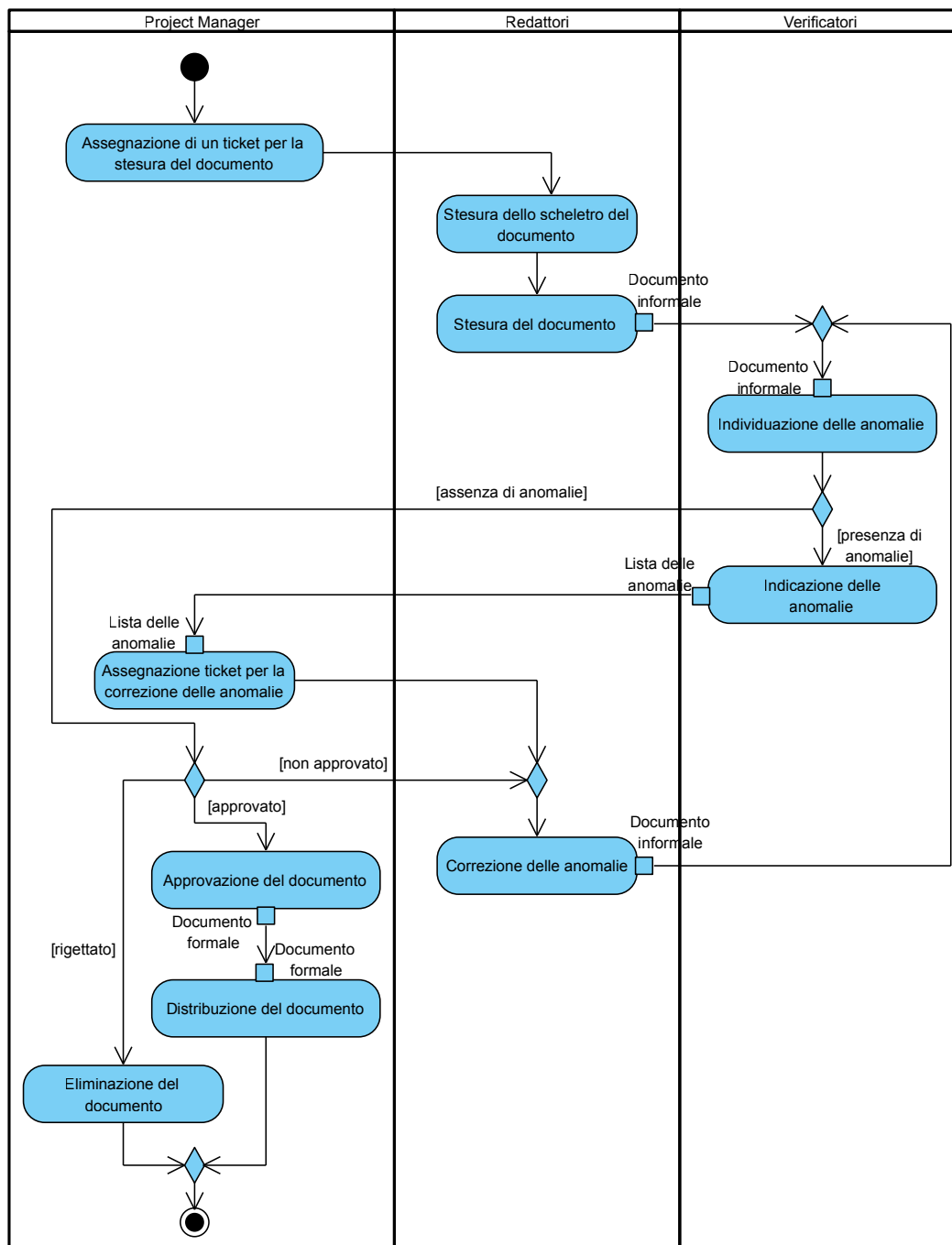


Figura 1: Diagramma di attività - formalizzazione dei documenti

### 3.1.2.2.1 Stile del testo

- **Grassetto:** questo stile dovrà essere applicato solo ai titoli e agli elementi di un elenco che riassumono il contenuto del paragrafo. Inoltre deve essere applicato alle parole che devono essere messe in risalto poiché sono il punto focale dell'argomento.
- **Corsivo:** lo stile corsivo verrà utilizzato per indicare percorsi di cartelle, file e nomi dei documenti, per riportare passaggi provenienti da fonti esterne, per gli elementi di un elenco di secondo livello e i termini del glossario.
- **Sottolineato:** questo stile verrà usato per far risaltare l'importanza della parola nella frase. Ad esempio, verrà usato con le particelle di negazione o con parole che stanno ad indicare l'obbligatorietà dell'azione.
- **Glossario:** questo stile è caratterizzato dalla marcatura della parola in stile corsivo e da una 'g' corsiva a pedice racchiusa tra '['. Tale stile si applica esclusivamente per indicare che il termine ha una corrispondenza nel glossario<sup>12</sup>.
- **Monospace:** tale stile viene adibito alla formattazione del testo contenente parti di codice o di comandi.
- **Maiuscolo:** l'utilizzo di parole le cui lettere sono tutte maiuscole dovrà limitarsi solo agli acronimi.
- **Colori:** l'utilizzo di differenti colori per il testo è da limitarsi solamente nel documento "*Definizione di Prodotto v2.00*". Tali verranno utilizzati per marcare maggiormente i nomi e i tipi degli attributi, e i nomi, i parametri e i tipi di ritorno dei metodi.  
In particolare, verrà seguito questo stile:
  - **VerdeOlive:** questo colore verrà utilizzato per il nome degli attributi;
  - **VerdeOliveScuro:** questo colore verrà utilizzato per il tipo degli attributi;
  - **VerdeScuro:** questo colore verrà utilizzato per il nome dei metodi;
  - **VerdePrimaveraScuro:** questo colore verrà utilizzato per i parametri dei metodi;
  - **VerdeMarinoMedio:** questo colore verrà utilizzato per il tipo di ritorno dei metodi.

### 3.1.2.2.2 Punteggiatura

- **Punteggiatura:** deve essere utilizzata in modo attento per raggruppare parti di frase dal senso coeso e coerente. L'utilizzo del punto è richiesto per terminare un concetto e passare alla discussione di un altro argomento affine.
- **Spazi:** nessun segno di punteggiatura verrà anticipato da spazi, a meno delle parentesi aperte.

---

<sup>12</sup>Non si considera l'utilizzo di tale stile nelle occorrenze valide presenti nei titoli, nei comandi e nei percorsi.

- **Maiuscolo:** le lettere iniziali maiuscole si useranno solo dopo segni di punteggiatura quali il punto, il punto di domanda e il punto esclamativo. Fanno ovviamente eccezione i nomi propri.
- **Parentesi:** le uniche parentesi che possono essere presenti all'interno di una frase sono quelle tonde. Possono essere impiegate ogni qual volta si vuole fornire un sinonimo del termine, oppure un esempio. Le quadre conterranno un determinato standard ISO oppure uno stato relativo ad un *ticket*<sub>[g]</sub>.
- **Apici:** gli apici singoli ‘ ’ dovranno essere utilizzati per racchiudere tra di essi un singolo carattere o segno.
- **Virgolette:** le doppie virgolette semplici “ ” verranno impiegate per racchiudere i nomi dei file e nei comandi se necessarie. Verranno impiegate anche per indicare che un insieme di parole è costituito da un ordine preciso (esempio: “Reti di calcolatori”). Le virgolette caporali, ‘«’ e ‘»’, si dovranno ritrovare solamente quando si vuole indicare una citazione.
- **Numeri:** i numeri andranno formattati secondo lo standard [SI/ISO 31-0] (esempio: 1 234 567,89).
- **Slash:** il simbolo ‘/’ si userà per indicare l'esclusione o l'entrambe (esempio: il presidente e/o il segretario) oppure nei percorsi di cartelle, di indirizzi web e nel codice.

### 3.1.2.2.3 Composizione del testo

- **Elenchi:** la prima parola di un elenco deve sempre essere maiuscola tranne nei casi in cui l'elenco sia preceduto da ‘:’. Ogni elemento di un elenco deve terminare con il simbolo di punteggiatura ‘;’ solamente se ogni elemento non contiene frasi complesse. Se anche un solo elemento prevede una o più frasi dalla struttura complessa, si dovrà imporre il simbolo ‘.’ al termine di tutti gli elementi. In ogni caso, l'ultimo elemento di un elenco dovrà terminare con il simbolo ‘.’.
- **Paragrafi:** i paragrafi devono essere sempre rientranti di 1 cm a sinistra dal margine della pagina e devono essere sempre giustificati.
- **Note a piè pagina:** ogni nota a piè pagina dovrà avere la prima lettera necessariamente maiuscola, a meno che non si tratti di un acronimo e in tal caso dovrà essere rispettata la formattazione dello stesso, e terminare con il simbolo ‘.’. Nel caso in cui la nota a piè pagina dovesse essere ripetuta più volte di seguito, la si dovrà indicare un'unica volta e porre lo stesso indice per richiamare tale nota.
- **Esempi:** quando viene indicato un esempio, lo si deve fornire in linea con il testo, racchiuso tra parentesi tonde e usando il seguente formalismo:

(esempio: ...)

- **Orfani e vedove:** gli orfani e le vedove sono delle linee tipografiche incomplete che restano isolate alla fine o all'inizio di una pagina. Tali linee non dovranno mai essere presenti in alcun documento. Per risolvere tali problematiche dovranno esserci almeno due linee isolate.

#### 3.1.2.2.4 Formati ricorrenti

- **Path:** per gli indirizzi web e i percorsi di cartelle si utilizza il comando appositamente creato. Tale comando serve per formattare gli indirizzi e i percorsi ed è reperibile in "*Guida Comandi Latex v1.00*".
- **Date:** se non specificato in modo diverso, deve essere espressa seguendo il formalismo dello standard [ISO 8601]:

AAAA-MM-GG

dove:

- *AAAA*: rappresenta l'anno. Per rappresentarlo si usano esattamente quattro cifre.
- *MM*: rappresenta il mese. Per rappresentarlo si usano esattamente due cifre<sup>13</sup>.
- *GG*: rappresenta il giorno. Per rappresentarlo si usano esattamente due cifre<sup>13</sup>.
- **Orari:** se non specificato in modo diverso, devono essere espressi seguendo il formalismo dello standard [ISO 8601]:

HH:MM

dove:

- *HH*: rappresenta il numero di ore trascorse dalla mezzanotte. Per la rappresentazione vengono usate esattamente due cifre<sup>13</sup>.
- *MM*: rappresenta i minuti. Per la rappresentazione vengono usate esattamente due cifre<sup>13</sup>.
- **Luoghi:** per trattare univocamente la dicitura di un luogo ci si deve riferire al seguente formato:

città (PR) in indirizzo, numero civico

dove:

- *città*: individua la città del luogo.
- *PR*: rappresenta la provincia della città. Per la rappresentazione si usano esclusivamente le sue iniziali in maiuscolo (esempio: PD).

---

<sup>13</sup>Nel caso il numero potesse essere rappresentato con una singola cifra, si impone l'anteposizione di uno zero alla cifra.

- *indirizzo*: fa riferimento all'indirizzo, privo di numero civico.
- *numero civico*: rappresenta il numero civico esclusivo della segnatura 'n.'.

- **Nomi ricorrenti:**

- *ruoli di progetto*: viene deciso che i vari ruoli di progetto andranno formattati utilizzando la prima lettera maiuscola. Tale decisione vale per ogni parola che non sia una preposizione.
- *nomi dei documenti*: viene deciso che per i nomi dei documenti si userà lo stile corsivo, utilizzando la prima lettera maiuscola per ogni parola che non sia una preposizione. I nomi dei documenti saranno racchiusi tra doppie virgolette semplici. Inoltre, vi sarà l'indicazione della versione (esempio: "*Norme di Progetto v5.00*"). Se il documento non esiste o la versione prevista è ignota, questa non dovrà essere riportata.
- *nomi dei file*: nel caso in cui si utilizzi il nome di un file, senza riportare l'indirizzo completo, si usa una formattazione particolare. Essa prevede che il nome del file sia racchiuso tra doppie virgolette semplici e che sia in corsivo (esempio: "*Norme\_Di\_Progetto.tex*").
- *nomi propri*: l'utilizzo dei nomi propri deve sempre prevedere il cognome che precede il nome, mai il contrario.

- **Riferimenti**: per fare riferimento ad una sezione che fa parte dello stesso documento, si deve scrivere il numero che la identifica. Se il riferimento si trova in una nota a piè pagina, allora va inserito anche il nome della sezione riferita.

- **URL**: l'inserimento di un URL relativo ad un indirizzo web deve essere formattato mediante il colore blu.

- **Sigle**: le sigle dei documenti dovranno essere usate solo nei diagrammi o nelle tabelle (con lo scopo di risparmiare spazio). Tali sigle devono essere scritte secondo il seguente formalismo:

- AdR ad indicare il documento "*Analisi dei Requisiti v6.00*";
- PdP ad indicare il documento "*Piano di Progetto v5.00*";
- PdQ ad indicare il documento "*Piano di Qualifica v5.00*";
- NdP ad indicare il documento "*Norme di Progetto v5.00*";
- Gl ad indicare il documento "*Glossario v2.00*";
- ST ad indicare il documento "*Specifiche Tecnica v3.00*".

### 3.1.2.3 Componenti grafiche





### 3.1.2.3.1 Immagini

Per aumentare la qualità nella lettura dei documenti si prevede che tutte le immagini inserite siano nel formato  $PDF_{|g|}$ . Nel caso in cui non fosse possibile impiegare l'utilizzo del formato PDF, è preferibile adottare il formato  $PNG_{|g|}$ .

Tutte le immagini devono essere corredate da una breve didascalia che ne descriva il contenuto. Inoltre si deve fornire un numero progressivo ad ogni immagine nel documento per permetterne la rintracciabilità. L'insieme di tutte le immagini presenti in un documento è riepilogato nella lista delle figure.

### 3.1.2.3.2 Tabelle

Si vuole migliorare la rintracciabilità delle tabelle e renderne più semplice la lettura. Per fare ciò, viene richiesto di rispettare le caratteristiche proposte qui di seguito:

- **titoli:** i titoli delle colonne devono essere formattati con il colore bianco e in grassetto.
- **intestazione:** l'intestazione contiene i titoli delle colonne. Essa deve prevedere il colore di sfondo  che corrisponde alla notazione in esadecimale 38A9A4.
- **intestazioni di secondo livello:** tali intestazioni contengono i titoli di secondo livello. Esse devono prevedere il colore di sfondo  che corrisponde alla notazione in esadecimale 8CD5DB. La presenza di intestazioni di secondo livello comporterà uno spostamento del contenuto. Lo spostamento sarà di 0,5 cm verso l'interno e riguarderà le righe successive.
- **righe:** le righe devono presentare colori alternati per avere una separazione più netta. Le righe in posizione dispari adotteranno il colore  che corrisponde alla notazione in esadecimale E1F4F3. Le righe in posizione pari adotteranno il colore  che corrisponde alla notazione in esadecimale B8E7E4.
- **bordi:** dovrà essere presente una linea bianca a separare le colonne.

Ogni tabella deve essere corredata da una didascalia che ne esplicita lo scopo e il contenuto. Inoltre, deve essere provvista di numero progressivo. Esso servirà per identificarla in modo univoco nel documento e per permettere una più facile rintracciabilità. L'insieme di tutte le tabelle verrà indicato nella lista delle tabelle per renderne più facile l'accesso nel documento. Allo scopo di semplificarne la lettura, si precisa che, nel caso si necessiti di indicare il valore 0 come numero, si preferisce il simbolo '-'.

### 3.1.2.4 Formattazione dei documenti

#### 3.1.2.4.1 Frontespizio

La prima pagina di ogni documento dovrà essere il frontespizio costituito dal seguente ordine di elementi:

1. **nome del progetto:** centrato e con una dimensione pari a 32 pt;
2. **nome del team:** centrato;
3. **email del team:** centrata, corsiva e con una dimensione pari a 16 pt;

4. **nome del documento:** centrato, in maiuscoletto, corredato dalla versione e con una dimensione pari a 24 pt;
5. **informazioni generali del documento:**
  - (a) *nome del documento*;
  - (b) *versione del documento*;
  - (c) *data di redazione*: deve essere indicata secondo il formato [ISO 8601];
  - (d) *redazione*: elenco in ordine alfabetico dei redattori del documento;
  - (e) *verifica*: elenco in ordine alfabetico dei Verificatori del documento;
  - (f) *approvazione*: viene indicato il soggetto responsabile di aver approvato il documento;
  - (g) *uso*: interno o esterno;
  - (h) *distribuzione*: elenco in ordine alfabetico dei soggetti a cui verrà distribuito il documento in oggetto.
6. **sommario:** brevissimo riassunto indicante lo scopo del documento.

#### 3.1.2.4.2 Diario delle modifiche

Il diario delle modifiche ha lo scopo di riepilogare l'elenco delle modifiche apportate al documento. Deve essere la pagina immediatamente successiva al frontespizio del documento. Il diario è una tabella ordinata in modo decrescente secondo la data della modifica e, di conseguenza, il numero di versione. Si adotta questo stile per focalizzare gli ultimi cambiamenti poiché trattati dalle prime righe della tabella.

Ogni riga del diario dovrà contenere i seguenti elementi nell'esatto ordine in cui sono indicati:

1. **informazione sulla modifica:** si tratta di una breve descrizione delle modifiche apportate.
2. **autore:** il soggetto responsabile delle modifiche specificate al punto precedente.
3. **data della modifica:** individua il giorno in cui il cambiamento è stato apportato.
4. **versione:** il numero di versione assegnato al documento. Tale numero viene assegnato in base alle norme previste nella sottosezione 3.1.2.7 relativa al versionamento.

#### 3.1.2.4.3 Indici

Gli indici hanno lo scopo di riepilogare e dare una visione macroscopica della struttura del documento. Permettono quindi di rintracciare i contenuti tramite una gerarchia. Tale gerarchia è basata sul livello delle sezioni.

Ogni documento, esclusi i verbali, dovrà essere corredato dall'indice dei contenuti. Esso sarà posizionato dopo il diario delle modifiche. Se sono presenti tabelle o immagini all'interno del documento, l'indice dei contenuti sarà seguito dalla lista delle tabelle e poi dalla lista delle figure.



#### 3.1.2.4.4 Intero documento

In ogni pagina, ad esclusione del frontespizio, dovranno comparire i seguenti elementi:

- **logo del team:** dovrà comparire, nell'intestazione della pagina, il logo del gruppo ProTech. Tale logo sarà posizionato a sinistra nelle pagine di numerazione dispari e a destra nelle pagine di numerazione pari.
- **sezione corrente:** il numero e il nome della sezione in cui ci si trova dovranno essere presenti nell'intestazione. Tali elementi saranno posizionati a sinistra nelle pagine di numerazione pari e a destra nelle pagine di numerazione dispari.
- **nome del documento:** a piè di pagina, a sinistra, deve apparire il nome del documento completo di versione.
- **numero di pagina:** a piè di pagina, a destra, deve comparire il numero della pagina espresso in relazione al numero totale di pagine.

Tutte le pagine avranno margine superiore pari a 2 cm, inferiore pari a 3,5 cm, destro e sinistro corrispondenti a 3 cm.

#### 3.1.2.5 Tipi di documenti

##### 3.1.2.5.1 Verbalì

I verbalì vengono prodotti dal soggetto incaricato alla loro stesura in occasione di incontri tra i membri del team e/o gli esterni. Per essi è prevista un'unica stesura. Tale scelta è motivata dal fatto che apportare modifiche implica una modifica delle decisioni prese in modo retroattivo.

I verbalì dovranno essere denominati secondo il seguente criterio:

Verbale\_data

dove la data<sup>14</sup> è relativa alla data dell'incontro e non alla stesura del documento.

Gli elementi del verbale devono sottostare al seguente ordine:

1. **data**<sup>14</sup>: come per il nome, anche qui si riferisce alla data dell'incontro.
2. **ordine del giorno:** indica che il coordinatore ha convocato regolarmente la riunione del gruppo ProTech alla data, alle ore e al luogo stabilito<sup>14</sup>.
3. **registrazione:** la sezione indica che il segretario procede con la verifica dei partecipanti. Il segretario inoltre redige l'elenco di questi in ordine alfabetico.
4. **approvazione del verbale:** atto del segretario che indica la lettura e l'approvazione del verbale. Il verbale riguarda la precedente riunione; fa eccezione la prima.

---

<sup>14</sup>Per la formattazione della data, dell'ora e del luogo si fa riferimento alla sottosezione 3.1.2.2.4 Formati ricorrenti.

5. **domande e risposte:** la sezione è alternativa alle due successive. Quando l'incontro coinvolge il proponente, verranno riportate in questa sezione le domande poste e le risposte date.
6. **attività approvate:** la sezione contiene l'elenco delle attività di cui si è discusso e che sono state approvate durante la riunione. Tale elenco comprende l'esito delle stesse.
7. **questioni in sospeso:** la sezione raggruppa le questioni lasciate in sospeso durante la riunione. In tale sezione vi sono anche le questioni che sono state rimandate a decisione futura.
8. **aggiornamento:** indica che il coordinatore ha aggiornato la riunione alle ore in cui è terminata la stessa.
9. **proponente:** indica il soggetto che ha proposto la riunione.
10. **approvato:** individua il Project Manager che ha approvato il verbale.

#### **3.1.2.5.2 Documenti informali**

Tutti i documenti saranno da ritenersi informali fino all'approvazione da parte del Project Manager. Egli potrà eventualmente richiederne una revisione. L'uso dei documenti informali è da considerarsi esclusivamente interno al team. Tale utilizzo è circoscritto alla sola redazione di tali documenti.

Per la denominazione dei file dei documenti si rimanda alla sottosezione 4.1.3.4.1.

#### **3.1.2.5.3 Documenti formali**

Una volta approvati dal Project Manager, i documenti si riterranno formali e adatti per essere distribuiti. Solo tali file potranno essere forniti ai soggetti presenti nella lista di distribuzione. Ogni volta che un documento formale verrà modificato, la nuova versione è da considerarsi non formale. Rimarrà tale fino alla sua successiva approvazione da parte del Project Manager. Sarà quindi trattata al pari di un documento informale.

#### **3.1.2.5.4 Glossario**

Il glossario conterrà alcune parole presenti negli altri documenti. Esse possono far parte del contesto di applicazione o trattano di termini che possono generare ambiguità d'interpretazione. I termini sono raggruppati in base alla prima lettera e ordinati secondo un criterio lessicografico. Tali termini devono essere corredati da una descrizione concisa e che non generi ulteriori equivoci.

I termini verranno inseriti nel glossario in modo parallelo alla stesura degli altri documenti, in modo tale da limitare errori umani. È consentito che venga inserito un termine inizialmente privo di definizione. Tale modalità è da preferirsi piuttosto che rimandare l'inserimento del termine nel glossario.

#### **3.1.2.6 Intestazione file di documentazione**

Ogni documento deve possedere la seguente intestazione:

`%FILE: Nome_Del_File.tex`

```
%PERCORSO: /Percorso/Del/Documento/Nome_Del_File/  
%DATA CREAZIONE: GG Mese AAAA  
%AUTORE: ProTech  
%EMAIL: protech.@gmail.com  
%  
%Questo file è proprietà del gruppo ProTech, viene rilasciato sotto licenza Apache  
v2.  
%  
%DIARIO DELLE MODIFICHE: presente nel file, se ne rimanda lì l'aggiornamento  
e la visione.  
% % % %
```

Il nome del file sarà identico al nome del documento, scritto secondo le norme<sup>15</sup>.

Il percorso utilizzerà  $\$/aSgad/\$$  come cartella di  $root_{|g|}$ . L'ultima cartella avrà lo stesso nome del documento, scritto secondo le norme<sup>15</sup>.

Un esempio della data scritta nel formato corretto è: 25 Dicembre 2013.

Gli ultimi quattro simboli % indicano il termine dell'intestazione. Essi sono necessari per il corretto funzionamento di alcuni script in fase di *pre-compilazione*<sub>|g|</sub>.

### 3.1.2.7 Versionamento dei documenti

#### 3.1.2.7.1 Avanzamento di versione documenti

Ogni volta che si inserisce una modifica nell'omonimo diario, va assegnato un nuovo numero di versione, in modo coerente con le seguenti regole:

- il numero di versione è indicato tramite il carattere 'v' seguito da un numero, da un punto e altri due numeri;
- quando verrà creato un documento avrà necessariamente il numero di versione 'v0.01';
- l'approvazione incrementa di un'unità il primo indice e imposta a '00' il secondo indice;
- una qualunque modifica lascia invariato il primo indice e incrementa di un'unità il secondo indice.

### 3.1.3 Strumenti

#### 3.1.3.1 LaTeX

Per la stesura dei documenti si è scelto di utilizzare il *linguaggio di markup*<sub>|g|</sub>  $\text{\LaTeX}$ . Il motivo principale che ha portato a questa scelta è la facilità di separazione tra contenuto e formattazione. Con  $\text{\LaTeX}$  è possibile definire l'aspetto delle pagine in un file template condiviso da tutti i documenti.

$\text{\TeX}$ studio versione 2.6.6 viene utilizzato come editor di testo  $\text{\LaTeX}$ . La compilazione in formato ".pdf" avviene per mezzo di PdfLatex.

---

<sup>15</sup>Formattata secondo le indicazioni previste in questo documento nella sottosezione 4.1.3.4.1 Norme sui nomi dei file.

### 3.1.3.2 Visual Paradigm

Per la creazione di diagrammi UML il team utilizza il software “Visual Paradigm for UML Community Edition”, versione 11.0. Nonostante la corposità del programma, prevale la semplicità di utilizzo dei suoi strumenti e la qualità estetica offerta. Inoltre Visual Paradigm permette l’esportazione in formato  $SVG_{|g|}$ . Tale formato è in grado di visualizzare oggetti di grafica vettoriale, pertanto è in grado anche di gestire immagini scalabili dimensionalmente.

### 3.1.3.3 Script

Per semplificare ed automatizzare il lavoro sono stati creati alcuni script. Essi sono stati scritti in Java per renderli portabili, o in  $PHP_{|g|}$  se relativi all’applicativo RACheL. La classe che contiene i primi, pronta all’utilizzo, si chiama `prompt` ed è accessibile nel repository al percorso `$/aSgad/Supporto/Script/Prompt/bin$`.

Gli script relativi alla documentazione vengono richiamati con i pulsanti del menu rapido di  $TeXstudio$ , ridefiniti secondo le istruzioni date dall’Amministratore.

Gli script che terminano con la compilazione del documento trovano e convertono automaticamente tutti i caratteri accentati in segnatura standard (esempio: ‘è’ diventerà `\{e}`).

Le funzionalità messe a disposizione sono le seguenti:

- **compilazione singolo file “.tex” per n volte:** lo script compila il file sorgente di  $LaTeX$  tante volte quante sono specificate nell’ultimo parametro. Lo script cercherà il file all’interno del percorso: `$/aSgad/Documenti/Documenti_In_Redazione$`. Si occuperà inoltre di cancellare i file di scarto che la compilazione produce.  
Lo script viene avviato tramite il comando `prompt -s nomefile.tex nCompilazioni`.
- **compilazione di tutti i file “.tex” per n volte:** lo script compila tutti i file sorgente di  $LaTeX$  presenti nel percorso indicato. La compilazione avviene tante volte quante ne sono specificate nell’ultimo parametro. Sono compresi nella compilazione anche tutti i file nelle sottocartelle. La compilazione dei file verrà lanciata concorrentemente su più  $thread_{|g|}$ . Lo script si occuperà inoltre di cancellare i file di scarto prodotti dalla compilazione. Se il percorso input è la stringa “def” allora lo script compilerà tutti i sorgenti  $LaTeX$  contenuti a partire dall’indirizzo: `$/aSgad/Documenti/Documenti_In_Redazione/$`.  
Lo script viene avviato tramite il comando `prompt -all percorso nCompilazioni`.
- **correzione ortografica su un file di estensione “.tex”:** lancia  $Hunspell_{|g|}$  sul file preso in input con i parametri corretti per l’uso del dizionario personalizzato. Esso elabora il file per renderlo compatibile con Hunspell. Subito dopo il ritorno di Hunspell il file in input viene riconvertito al formato  $TeX_{|g|}$ .  
Lo script viene avviato tramite il comando `prompt -h percorso/nomefile.tex`.
- **calcolo dell’indice Gulpease su un file di estensione “.tex”:** lo script calcola l’indice  $Gulpease_{|g|}$  sul file in input e produce un file “*Gulpease.txt*”, nella stessa directory, contenente i risultati.  
Lo script viene avviato tramite il comando `prompt -g percorso/nomefile.tex`.

- **avvio di procedura di glossarizzazione su un file di estensione “.tex”**: lo script avvia in una shell una procedura che permette di scegliere quali parole segnalare come appartenenti al glossario. Si è vista la necessità di un approccio parzialmente manuale in quanto alcune parole possono assumere un significato diverso a seconda del contesto, e non per forza la prima occorrenza di un termine presente nel glossario assume lo stesso significato indicato.

Lo script viene avviato tramite il comando `prompt -d percorso/nomefile.tex`.

- **aggiornamento dei riferimenti locali a documenti esterni su un file di estensione “.tex”**: lo script ricerca nel documento tutti i riferimenti relativi ad altri documenti, aggiornandone la versione nominata all’ultima presente per tale documento.

Lo script viene avviato tramite il comando `prompt -il percorso/nomefile.tex`.

- **confronto e aggiornamento del database a partire da un file di estensione “.xml”**: a partire da un diagramma delle classi, lo script si occupa di segnalare, connettendosi al database, quali elementi sono già presenti e quali elementi sono mancanti, dando la possibilità di aggiungere rapidamente relazioni, dipendenze, attributi e metodi assenti.

Lo script viene avviato tramite una funzionalità offerta dall’applicativo RACheL.

## 3.2 Processo di verifica

### 3.2.1 Attività

#### 3.2.1.1 Analisi

##### 3.2.1.1.1 Analisi statica

L’analisi statica è una tecnica di verifica applicabile sia ai documenti sia al codice che verrà effettuata durante tutto lo sviluppo del sistema. Serve a trovare le anomalie, che verranno trattate come specificato in questo documento. Può essere applicata nei due modi seguenti.

- **Walkthrough**:

Consiste in una lettura del documento/codice cercando errori ed anomalie a largo spettro senza un’idea precisa di quali tipi errori sarà possibile trovare. Ogni difetto rilevato sarà discusso con gli autori allo scopo di evitare incomprensioni e concordare sulle modifiche necessarie.

Il *walkthrough*<sub>[g]</sub> è indispensabile durante lo sviluppo iniziale, quando non si possiede ancora una chiara visione sui possibili errori. Usando più volte questa tecnica sarà possibile stilare una “lista di controllo”<sup>16</sup> dove verranno archiviati gli errori più spesso rilevati.

Inizialmente le attività di verifica utilizzeranno in prevalenza la tecnica *walkthrough*. Non appena i Verificatori avranno stilato una lista di controllo sufficiente, si passerà sempre più all’uso della tecnica dell’*inspection*<sub>[g]</sub>.

---

<sup>16</sup>La lista di controllo quando disponibile sarà allegata in Appendice al documento “*Norme di Progetto v5.00*”.

- **Inspection:**

L'inspection si basa sulla lettura mirata dei documenti/codice. Durante tale lettura si cercano gli errori segnalati nella lista di controllo. Progressivamente con l'acquisizione di esperienza la lista di controllo verrà estesa. Questo renderà l'inspection sempre più efficace.

### 3.2.1.1.2 Analisi dinamica

L'analisi dinamica è applicabile solo a componenti software e viene svolta tramite l'esecuzione di test su essi. Essi ne verificheranno il funzionamento, e in caso di anomalie aiuteranno ad identificarle.

Per garantire risultati attendibili è necessario che i test siano ripetibili. Quindi dato lo stesso input, lo stesso test deve produrre lo stesso output se eseguito nello stesso ambiente. Solo un test con queste caratteristiche è in grado di riscontrare problemi e verificare la correttezza del prodotto software.

Per ogni test deve essere definito:

- **ambiente:** consiste nel sistema hardware e software sui quali è pianificata l'esecuzione del test sul prodotto. Va inoltre specificato uno stato iniziale dal quale il test deve partire.
- **specifica:** consiste nella specifica degli input e degli output attesi.
- **procedure:** consiste nella specifica di ulteriori istruzioni su come va eseguito il test e su come vanno interpretati i risultati.

Sono definiti nelle prossime sezioni i tipi di test che il team effettuerà sul prodotto software in sviluppo.

### 3.2.1.2 Test

#### 3.2.1.2.1 Test di unità

Il test di unità verifica che ogni singola unità<sup>17</sup> di prodotto software funzioni correttamente.

Attraverso questo test si verificherà la correttezza di tutti i moduli base che compongono i software andando a limitare gli errori di implementazione.

I test di unità possono essere identificati grazie alla seguente sintassi:

TU[Codice test]

#### 3.2.1.2.2 Test di integrazione

Il test di integrazione verifica che due o più moduli precedentemente verificati una volta assemblati insieme funzionino come previsto. Aiuta inoltre a rilevare i difetti residui dei moduli non scovati durante i test precedenti.

Questo test verifica inoltre la collaborazione dei moduli prodotti con componenti

---

<sup>17</sup>Per unità si intende una sezione di software abbastanza piccola da poter assegnare lo sviluppo ad un singolo Programmatore.

software esterne come *framework*<sub>|g|</sub> o *librerie*<sub>|g|</sub>.

Può essere che vengano inserite delle componenti “fittizie” che verranno impiegate al posto di moduli non ancora pronti per poter effettuare una analisi sul risultato di certi test.

I test di integrazione possono essere identificati grazie alla seguente sintassi:

TI[Identificativo del componente]

Tale identificativo corrisponde al componente i cui elementi sono integrati.

#### 3.2.1.2.3 Test di sistema

I test di sistema consistono nella validazione dei prodotti software. Questo test viene eseguito quando si ritiene che il prodotto sia giunto ad una versione definitiva. Viene quindi verificata la completa copertura dei requisiti da parte del prodotto.

I test di sistema possono essere identificati grazie alla seguente sintassi:

TS[Tipo Requisito][Codice Requisito]

Il tipo e il codice si riferiscono al requisito di cui verrà testato il soddisfacimento.

#### 3.2.1.2.4 Test di regressione

Questo test viene eseguito subito dopo che una componente viene modificata. Consiste nel rieseguire tutti i test per verificare che dopo le modifiche il resto dei moduli continuino a funzionare in modo corretto.

Il tracciamento aiuta a capire quali sono i test da ripetere (di ogni tipo) poiché potenzialmente a rischio in caso di modifica.

#### 3.2.1.2.5 Test di validazione

Coincide con il collaudo del software in presenza del proponente. Se tale test ha esito positivo il prodotto sarà considerato abbastanza maturo da permetterne il rilascio.

I test di validazione possono essere identificati grazie alla seguente sintassi:

TV[Tipo Requisito][Codice Requisito]

Il tipo e il codice si riferiscono al requisito di cui verrà testato il soddisfacimento.

#### 3.2.1.3 Tracciamento

Sarà compito dei Verificatori controllare la corrispondenza di ogni requisito con una o più fonti. In caso di anomalie bisogna comportarsi come descritto nella sottosezione 3.2.2.1.

### 3.2.2 Procedure

#### 3.2.2.1 Procedura per la gestione delle anomalie

Al completamento di un ticket, il Verificatore dovrà procedere alla verifica dell’operato. Qualora individuasse un’anomalia<sup>18</sup>, dovrà registrarla in una lista delle anomalie individuate. Si chiede di adottare una forma standard per poter rintracciare qualsiasi tipo di errore ed individuare quali siano quelli più comuni. In seguito, verranno riportati nella “lista di controllo”<sup>19</sup> per favorire una tecnica di ispezione mirata.

---

<sup>18</sup>I vari casi individuati come anomalie sono descritti nel documento “*Piano di Qualifica v5.00*”.

<sup>19</sup>La lista di controllo viene gestita tramite Tracker, ulteriori informazioni sono disponibili nella sottosezione 3.2.3.5 Tracker.

Al termine dell'attività di verifica verranno creati dal Project Manager dei ticket. Tali ticket verranno assegnati ai responsabili delle anomalie, per la risoluzione delle stesse. Della lista delle anomalie verrà fornita una visione della stessa che è di competenza esclusiva del Project Manager. Le anomalie individuate dovranno essere corredate da una breve descrizione se necessario. Di seguito si fornisce la sequenza di passi previsti dalla procedura:

1. il Verificatore, mentre svolge l'attività di verifica di un documento o di una parte di codice, individua un insieme di anomalie.
2. il Verificatore inserisce le anomalie raccolte in una lista delle anomalie riscontrate. Ne indica i file in cui è stata riscontrata l'anomalia, l'ubicazione all'interno del file e il tipo.
3. il Project Manager assegna un ticket ai soggetti responsabili delle anomalie raccolte. Egli indicherà, per ogni responsabile, quali sono le anomalie da correggere.

La precedente procedura viene riassunta dal diagramma delle attività indicato in figura 2.

### 3.2.3 Strumenti

#### 3.2.3.1 Correzione ortografica automatica

La correzione ortografica avviene in tempo reale durante la scrittura nell'editor  $\text{\TeX}$ studio. Si obbliga a posizionare i file "*it\_IT.aff*", "*it\_IT.dic*" e "*th\_it\_IT.dat*" nella directory<sub>[g]</sub> di  $\text{\TeX}$ studio  $\$ \text{\TeX}studio/dictionaries \$$ . Tali file sono stati distribuiti ai membri del team dall'Amministratore. I file vanno quindi abilitati dal menu di configurazione dell'editor nel pannello "Configure  $\text{\TeX}$ studio". Il limite di questo tipo di correzione è che gli errori possono essere facilmente non visti e accidentalmente ignorati. Ulteriore problema è che il correttore automatico segna come errate le parole che usano gli accenti con la marcatura  $\text{\LaTeX}$ . Perciò si è scelto di sfruttare parallelamente il software Hunspell<sup>20</sup>.

#### 3.2.3.2 Hunspell

Supporto ulteriore alla correzione ortografica è dato da Hunspell 1.3.2. Dopo aver provato diversi tool, la scelta è ricaduta su Hunspell in quanto è l'unico a supportare le lettere accentate, che non sono state correttamente riconosciute da strumenti simili come *Aspell*<sub>[g]</sub>.

L'utilizzo di Hunspell viene chiamato tramite un pulsante nell'editor  $\text{\TeX}$ Studio. Tale pulsante avvia uno script personalizzato.

Lo script converte tutte le lettere accentate che sono in formato  $\text{\LaTeX}$ , in lettere accentate in codifica "ANSI-8859-1". Verrà poi avviato lo strumento di correzione ortografica sul documento. Esso indicherà all'utente tutti gli errori non ancora segnalati, grazie all'utilizzo di un vocabolario di progetto<sup>21</sup>. Infine riconverte tutte le lettere accentate nel formato iniziale.

---

<sup>20</sup>Si veda la sottosezione 3.2.3.2 Hunspell.

<sup>21</sup>In tale vocabolario verranno inserite le parole relative al progetto non presenti nel vocabolario italiano.



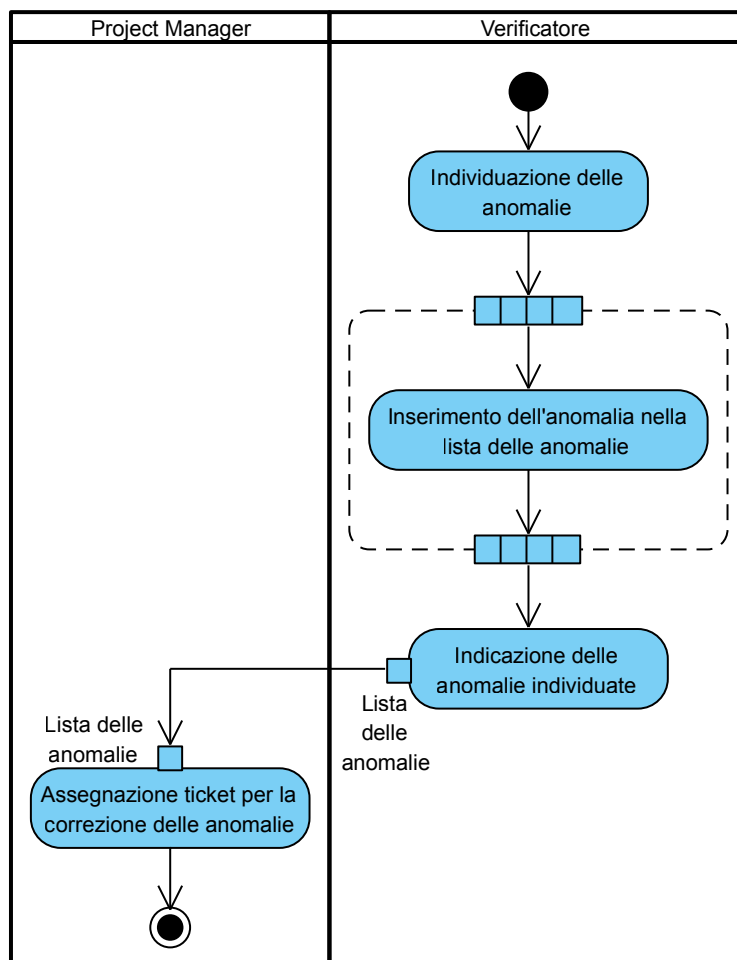


Figura 2: Diagramma di attività - procedura per la gestione delle anomalie

### 3.2.3.3 Calcolo Gulpease

Per calcolare la leggibilità di un documento si utilizza l'indice Gulpease. Uno degli script creati dal team<sup>22</sup> provvede al calcolo di tale indicatore.

### 3.2.3.4 Strumenti per la validazione W3C

Per l'applicativo web vengono utilizzati gli strumenti di validazione online <http://validator.w3.org> per le pagine web e <http://jigsaw.w3.org/css-validator/> per i fogli di stile.

### 3.2.3.5 Tracker

È stato creato questo strumento per semplificare il tracciamento degli errori. Tracker è una pagina web che permette di segnalare, in maniera immediata e semplice, gli errori riscontrati durante la verifica. Il Project Manager avrà, così, una veloce panoramica sugli errori presenti e potrà, in questo modo, assegnare con facilità i ticket di correzione.

#### 3.2.3.5.1 Metriche

Per il calcolo delle varie metriche vengono adottati strumenti diversi per i linguaggi adottati.

Per quanto riguarda il linguaggio JavaScript sono stati adottati i seguenti strumenti:

- RACheL fornisce le metriche quali il numero di attributi di una classe e il numero di parametri per metodo avendo già memorizzato tali dati.
- [www.jsmeter.info](http://www.jsmeter.info) fornisce il resto delle metriche ad esclusione del numero di chiamate innestate di metodi. Uno script realizzato dal gruppo si occupa di ottenere le metriche fornite da tale strumento per memorizzarle nell'applicativo RACheL.
- Infine è stato utilizzato lo strumento JSCover per calcolare la copertura del codice, questo strumento si avvale di un eseguibile Java che analizza i file sorgente JavaScript ed evidenzia quali *statement*<sub>[g]</sub> non vengono eseguiti durante un certo lasso di tempo prefissato.

Per quanto riguarda invece il linguaggio Scala sono stati adottati i seguenti strumenti:

- Per effettuare i test di unità è stato utilizzato il *toolkit*<sub>[g]</sub> ScalaTest. Questo è uno strumento molto utile per effettuare test di unità in un ambiente Java o Scala e viene facilmente utilizzato tramite una importazione dei package relativo all'interno del codice di test.
- Il calcolo delle metriche è stato effettuato attraverso diversi strumenti. In primo luogo, grazie alle funzionalità offerte dall'applicativo RACheL, il calcolo automatico relativo alle metriche di attributi e parametri era già stato effettuato precedentemente. Successivamente si è utilizzato uno script creato appositamente che, interfacciandosi allo strumento CLOC, calcolasse le metriche analizzando il codice sorgente.
- Per la copertura del codice è stato utilizzato lo strumento SCCT che permette di avere una buona visione d'insieme evidenziando la copertura generale e relativa. Per

---

<sup>22</sup>Si veda la sottosezione 3.1.3.3 Script.

quanto riguarda la copertura relativa permette di controllare con diversi gradi di granularità tra cui componente, classe e statement.

## 4 Processi organizzativi

### 4.1 Processo di gestione

#### 4.1.1 Attività

##### 4.1.1.1 Comunicazioni

###### 4.1.1.1.1 Comunicazioni interne

Le comunicazioni interne vengono gestite tramite un gruppo *Facebook*<sub>[g]</sub> privato. Tale gruppo è denominato “[SWE] ProTech” ed è accessibile ai soli membri del team.

Per comunicazioni rapide a tutti i membri del team, si utilizza l’indirizzo di posta elettronica:

protech.unipd@gmail.com

Il gruppo si avvale di un sistema di avvertimento che, tramite mail, avverte i diretti interessati alla creazione di un ticket. Le meccaniche e le norme relative a questo strumento di ticketing verranno trattate in seguito nella sezione 4.1.1.3.

Nel caso sia richiesto l’utilizzo di una chat o di una videoconferenza, verrà utilizzata l’applicazione *Skype*<sub>[g]</sub>.

###### 4.1.1.1.2 Comunicazioni esterne

Il Project Manager, dato che rappresenta il gruppo ProTech, mantiene i contatti con le varie componenti esterne al team. Per farlo, utilizza una apposita casella di posta elettronica:

pm.protech.unipd@gmail.com

Il Project Manager valuterà se è il caso di inoltrare tali mail agli altri membri del team tramite la *mailing list*<sub>[g]</sub> a cui si fa riferimento nella sottosezione 4.1.1.1.1.

##### 4.1.1.2 Gestione incontri

###### 4.1.1.2.1 Incontri interni

Il Project Manager ha il compito di fissare gli incontri interni. Scelta una data idonea all’incontro, dovrà avvertire i membri del team. Per farlo, utilizzerà la mailing list indicata nella sottosezione 4.1.1.1.1.

Per ogni nuovo incontro dovranno essere specificati la data, l’ora, il luogo, il proponente e la motivazione che lo hanno reso necessario. Tali informazioni dovranno essere rese disponibili con almeno tre giorni di anticipo. Vi possono essere casi eccezionali che richiedono un intervento tempestivo. Tali casi, come ad esempio una *milestone*<sub>[g]</sub> imminente, potranno non soddisfare tale prerequisite.

Ogni membro del team può richiedere un incontro interno al Project Manager. Egli, accertati i motivi e verificata la necessità, si preoccuperà di organizzare un eventuale incontro.

Ogni membro del team è tenuto a rispondere alla mail riguardante richieste di incontri. Tale mail sarà indirizzata unicamente al Project Manager. In tale mail egli dovrà confermare la sua presenza o giustificare, succintamente, la sua assenza. Nel caso in cui si voglia rettificare una vecchia mail, l'interessato dovrà scrivere una nuova mail al Project Manager. Egli dovrà prendere in considerazione solamente l'ultima mail recepita.

#### 4.1.1.2.2 Incontri esterni

È compito del Project Manager concordare, con il proponente o con i committenti, gli incontri esterni. Per farlo seguirà l'iter indicato nella sezione 4.1.1.1.2 Comunicazioni esterne.

Ogni membro del team può richiedere un incontro specificandone in breve il motivo al Project Manager. Egli, prima di prendere accordi con le parti esterne coinvolte, dovrà assicurarsi del consenso e della presenza di almeno due membri del team. Nel caso in cui non ci sia tale consenso, la proposta di incontro sarà bocciata. In caso contrario, il Project Manager dovrà contattare la parte esterna interessata per prendere accordi. Le informazioni circa l'incontro (data, ora, luogo) dovranno essere rese disponibili al team.

Sarà compito di uno dei presenti, delegato di volta in volta, redigere il verbale dell'incontro avvenuto.

#### 4.1.1.3 Ticketing

Il sistema di ticketing viene utilizzato per suddividere i *task*<sub>[g]</sub> tra tutti i componenti del team. Il sistema permette di visionare in ogni istante i compiti assegnati a ciascuno (completati, in corso e ancora da svolgere).

#### 4.1.1.4 Repository

Per svolgere in modo corretto i compiti all'interno del team, è necessario definire con assoluta precisione il luogo in cui si cercano e si salvano tutti i file relativi al progetto. Si è dunque scelto di usare un repository remoto del team.

Il servizio di hosting scelto permette con licenza "educational" di impostare la visibilità del repository ai soli utenti invitati. Tutti i membri del team sono quindi tenuti a registrarsi e a notificare all'Amministratore il nome utente. Egli provvederà quindi ad autorizzare i membri all'accesso al repository.

### 4.1.2 Procedure

#### 4.1.2.1 Procedura di assegnazione

L'assegnazione di ticket è delegata al Project Manager, che utilizzerà lo strumento *MyTinyToDo*<sub>[g]</sub>. Tale strumento è una particolare applicazione JavaScript personalizzata per gli scopi del team.

In questa applicazione i ticket vengono divisi in liste nominative che identificano il membro a cui viene affidato il ticket. La procedura di assegnazione di un ticket segue il diagramma di attività riportato in figura 3.

1. Il Project Manager sceglie a chi assegnare il ticket;
2. Ne indica l'ambito secondo le regole descritte nella sottosezione 4.1.3.1;

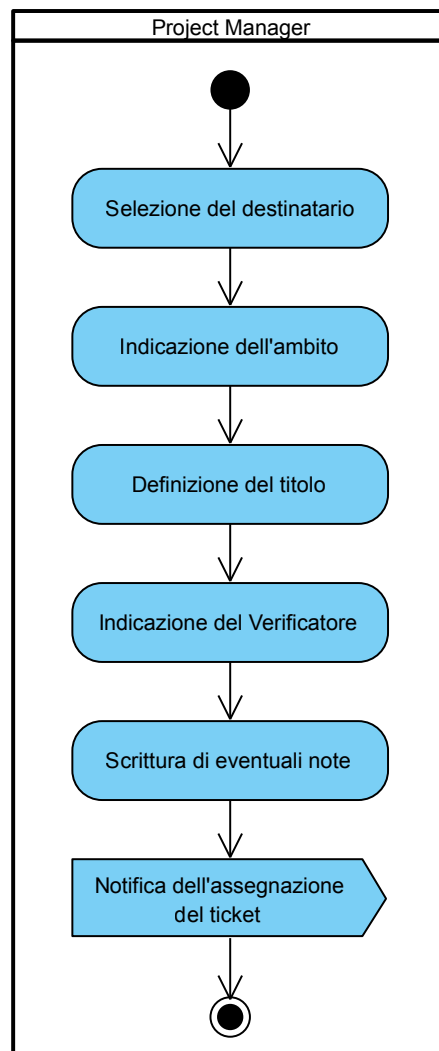


Figura 3: Diagramma di attività - assegnazione di un ticket

3. Ne definisce il titolo secondo le regole descritte nella sottosezione 4.1.3.1;
4. Specifica il Verificatore del ticket;
5. Scrive eventuali note atte a descrivere meglio il task da svolgere;
6. L'assegnatario del ticket riceve una notifica mail che lo informa del ticket assegnatogli.

#### 4.1.2.2 Procedura per la rilevazione dei rischi

Durante lo svolgersi del progetto, il Project Manager ha la responsabilità di individuare i rischi indicati nel “*Piano di Progetto v5.00*”.

Vi possono essere casi in cui si verificano delle problematiche non previste. In tali casi il Project Manager dovrà estendere l'analisi dei rischi e includervi la gestione delle nuove problematiche. Complessivamente la procedura prevede i seguenti passi:

1. rilevazione ed individuazione attiva dei rischi previsti e dei problemi non calcolati;
2. registrazione del riscontro effettivo dei rischi nel “*Piano di Progetto v5.00*”;
3. trattazione e pianificazione per la gestione dei nuovi rischi individuati;
4. ridefinizione e aggiornamento delle strategie in base alle necessità e alle previsioni.

#### 4.1.3 Norme

##### 4.1.3.1 Regole generali

Ogni ticket possiede una sintassi standard e deve essere conforme alle seguenti regole:

- un ticket deve rispettare il formato:

[Ambito] Titolo [Verificatore] [Stato]

In questo formato, “[Ambito]”, “[Verificatore]” e “[Stato]” sono rispettivamente i *tag*<sub>|g|</sub> di Ambito, Verifica e Stato.

- il nome del ticket deve riassumere brevemente le mansioni da completare.
- qualora il titolo non sia abbastanza esaustivo o si vogliano inserire delle note, si dovrà utilizzare il campo apposito “Note”.
- alla creazione del ticket, il Project Manager dovrà preoccuparsi di assegnare la verifica di tale compito ad un membro del team. Il nominativo del membro del team designato verrà quindi inserito nel titolo del ticket, racchiuso tra tag.
- la priorità di un ticket è determinata dalla sua data di scadenza. Di conseguenza, verrà data maggiore priorità a ticket con scadenza prossima alla data attuale. I ticket con una data di terminazione più lontana nel tempo avranno una priorità minore.

Da parte dei membri del team è invece richiesta una prassi di utilizzo riguardo ai ticket. Ciò aiuta il Project Manager a supervisionare le attività correnti. Vengono quindi elencate le norme sintattiche relative al titolo del ticket:

- i ticket di cui si avrà presa visione saranno identificati dal tag di stato “[Accettato]”.
- i ticket su cui si sta lavorando saranno identificati dal tag di stato “[In corso]”.
- i ticket interrotti per ricezione di ticket prioritari saranno identificati dal tag di stato “[Sospeso]”.
- i ticket interrotti per mancanza di risorse, input o problematiche, saranno identificati dal tag di stato “[In attesa]”. Tali ticket conterranno nel campo note una descrizione circa la motivazione dell’attesa.
- i ticket completati e che sono in attesa di verifica saranno identificati dal tag di stato “[Completato]”.
- i ticket verificati, quindi conclusi, saranno identificati dal tag di stato “[Verificato]”.

#### 4.1.3.2 Protocollo di utilizzo

La procedura di svolgimento di un ticket da parte dell’assegnatario segue il diagramma di attività riportato in figura 4.

1. Ogni membro, ricevuto almeno un ticket attivo, ne sceglierà uno prediligendo quelli a scadenza più imminente.
2. Ogni utente potrà avere al massimo un solo ticket con la dicitura “[In corso]” contemporaneamente.
3. Nel caso in cui un utente riceva un ticket a scadenza più imminente rispetto al ticket che in quel momento detiene il tag “[In corso]”, egli dovrà sospendere il ticket corrente e dovrà iniziare a lavorare al ticket prioritario.
4. Una volta concluso un ticket, il membro del team in questione dovrà attendere l’esito del Verificatore. Egli, nel caso in cui reputi concluso il ticket, dopo aver rispettato le suddette norme sintattiche, potrà spuntare la casella di controllo. In caso contrario, seguirà la prassi descritta nella sezione 3.2.2.1.

Il sistema è stato sviluppato in modo da avvertire i diretti interessati al verificarsi di determinati eventi. Questo sistema è diviso in tre momenti: inizialmente, alla creazione di un ticket, verrà avvisato il membro del team interessato; quando l’assegnatario avrà concluso il ticket, il Verificatore scelto verrà avvisato; infine, dopo che la verifica è stata completata, verrà avvisato il Project Manager del reale completamento del ticket.

Nel caso in cui si incontri una problematica non gestita da queste norme, si rimanda al giudizio dell’Amministratore.

#### 4.1.3.3 Ruoli di progetto

Lo sviluppo di un progetto prevede la collaborazione di individui a cui sono stati assegnati diversi ruoli. Tali ruoli rappresentano le omonime figure aziendali, indispensabili per il buon esito del progetto. Si garantisce che ogni componente del team ricoprirà tutti i ruoli almeno una volta. A tal proposito, potrebbero sorgere delle problematiche relative



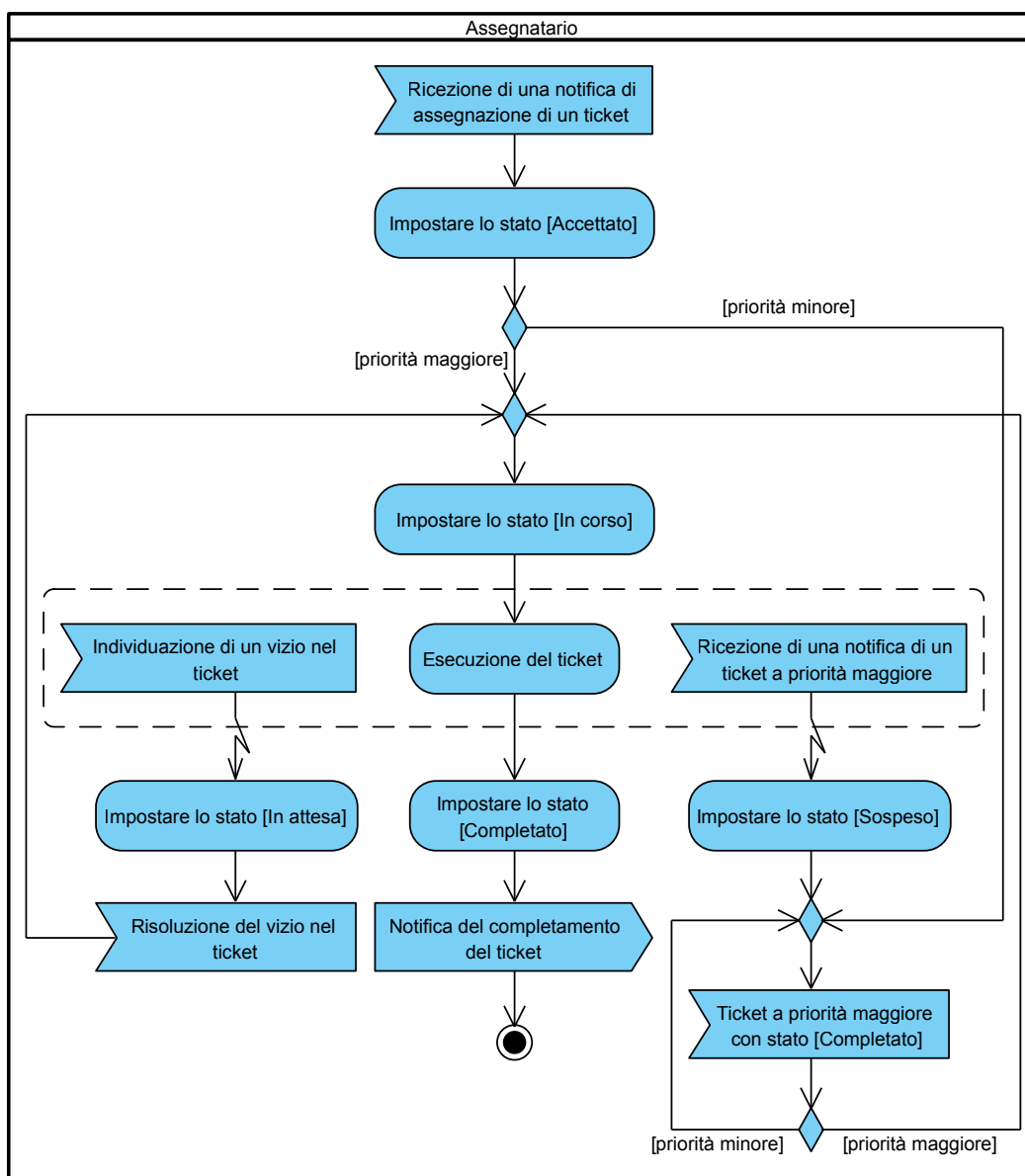


Figura 4: Diagramma di attività - svolgimento di un ticket

a conflitti d'interesse (esempio: lo stesso soggetto potrebbe ritrovarsi a ricoprire un ruolo e, in seguito, a verificare l'operato svolto in precedenza). Queste situazioni non dovranno mai presentarsi poiché potrebbero ledere la qualità del prodotto. È quindi importante che le attività vengano pianificate con attenzione e che ciascun membro del team assicuri l'esclusivo svolgimento del ruolo a lui assegnato. Sarà compito del Verificatore controllare che la pianificazione rispetti le considerazioni di cui sopra. Al sorgere di simili incongruenze, si provvederà a notificare il Project Manager che avrà la responsabilità di risolvere la questione.

Di seguito si discuterà dei vari ruoli, delle relative responsabilità e delle modalità operative.

#### 4.1.3.3.1 Project Manager

Il **Project Manager** rappresenta il progetto, in quanto accentra su di sé le responsabilità di scelta e di approvazione, e il team, poiché presenta al committente i risultati del lavoro svolto.

Detiene il potere decisionale, quindi la responsabilità in merito a:

- pianificazione, coordinamento e controllo delle attività;
- gestione e controllo delle risorse;
- analisi e gestione dei rischi;
- approvazione dei documenti;
- approvazione dell'offerta economica.

Di conseguenza, ha il compito di:

- assicurarsi che le attività di verifica e validazione vengano svolte sistematicamente riferendosi alle “*Norme di Progetto v5.00*”;
- garantire che vengano rispettati i ruoli e le competenze assegnate nel “*Piano di Progetto v5.00*”;
- garantire che non vi siano conflitti di interessi tra redattori e Verificatori.

#### 4.1.3.3.2 Amministratore

L'**Amministratore** è il responsabile dell'efficienza e del rendimento dell'ambiente di lavoro. Le mansioni che gli competono sono le seguenti:

- ricercare strumenti che possano automatizzare il maggior numero possibile di compiti e operazioni.
- gestire l'archiviazione e il versionamento della documentazione di progetto.
- controllare le versioni dei prodotti e gestire le loro configurazioni.
- fornire strumenti e procedure per la segnalazione ed il monitoraggio ai fini del controllo qualità.

- risolvere i problemi legati alle difficoltà di controllo e alla gestione delle risorse e dei processi. La risoluzione di tali problemi richiede l'adozione di strumenti adeguati.

L'Amministratore inoltre redige le “*Norme di Progetto v5.00*” nelle quali spiega e norma l'utilizzo degli strumenti, e redige la sezione del “*Piano di Qualifica v5.00*” in cui si descrivono strumenti e metodi di verifica.

#### 4.1.3.3.3 Analista

L'**Analista** è il responsabile delle attività di analisi. Le mansioni di questo ruolo sono le seguenti:

- comprendere appieno la natura e la complessità del problema;
- produrre una specifica di progetto che sia motivata in ogni suo punto e al tempo stesso comprensibile sia dal proponente, sia dal committente, sia dai Progettisti.

L'Analista, inoltre, redige il documento “*Analisi dei Requisiti v6.00*” e lo “*Studio di Fattibilità v1.00*”.

#### 4.1.3.3.4 Progettista

Il **Progettista** è il responsabile delle attività di progettazione. Le mansioni di questo ruolo sono le seguenti:

- effettuare scelte su aspetti progettuali e tecnologici che rendano il prodotto facilmente manutenibile in futuro;
- effettuare scelte su aspetti di progettazione che applichino al prodotto soluzioni note ed ottimizzate;
- produrre una soluzione comprensibile, attuabile e motivata.

Il Progettista redige anche la “*Specifica Tecnica v3.00*”, la “*Definizione di Prodotto*” e le sezioni relative alle metriche di verifica della programmazione del “*Piano di Qualifica v5.00*”.

#### 4.1.3.3.5 Programmatore

Il **Programmatore** è il responsabile delle attività di codifica e delle componenti di ausilio necessarie per effettuare le prove di verifica. Le mansioni di questo ruolo sono le seguenti:

- scrivere codice che sia documentato, versionato, manutenibile e che rispetti le metriche stabilite per la scrittura del codice;
- implementare in maniera rigorosa le soluzioni descritte dal Progettista;
- implementare i test sul codice prodotto, necessari per le prove di verifica.

Il Programmatore inoltre, redige il “*Manuale Utente*”.

#### 4.1.3.3.6 Verificatore

Il **Verificatore** è il responsabile delle attività di verifica. Le mansioni di questo ruolo sono le seguenti:

- controllare la conformità di ogni stadio del ciclo di vita del prodotto;
- garantire che l'attuazione delle attività sia conforme alle norme stabilite.

Il Verificatore inoltre, redige la sezione del “*Piano di Qualifica v5.00*” che illustra l'esito e la completezza delle verifiche e delle prove effettuate.

#### 4.1.3.4 Repository

##### 4.1.3.4.1 Norme sui nomi dei file

I membri sono tenuti ad utilizzare la seguenti norme di denominazione dei file presenti nel repository:

- per rendere più agevole il reperimento dei file, ognuno di essi dovrà essere denominato secondo il seguente formalismo:

Nome\_Del\_File.ext

dove “Nome\_Del\_File” rappresenta il nome del file. Nel caso il nome fosse composto da più parole, la prima lettera di ogni parola deve essere maiuscola e non devono esserci spazi ma underscore ‘\_’ per separare le parole. Nel caso in cui si tratti di file relativi alla documentazione con estensione “.pdf”, il nome del file dovrà terminare con il numero di versione separato dal nome mediante l'underscore ‘\_’ e preceduto da ‘v’. Non sono ammesse le lettere accentate (esempio: Studio\_Di\_Fattibilita\_v1.23.pdf).

##### 4.1.3.4.2 Struttura del repository

I membri sono tenuti a rispettare le seguenti norme di organizzazione dei file nel repository:

- la root del repository contiene le sole tre cartelle:
  1. **Codice:** contenente il codice del progetto;
  2. **Documenti:** contenente le seguenti cartelle:
    - (a) una cartella per ogni documento diverso, il nome della cartella deve essere lo stesso del documento (senza versione). Se nel documento sono presenti immagini, queste andranno inserite in una sottocartella di nome “Immagini”. Se presenti diagrammi essi andranno inseriti in una sottocartella di nome “Diagrammi”;
    - (b) **Verbali:** contenente le seguenti cartelle:
      - i. una cartella per ogni verbale e il suo nome deve essere lo stesso del documento (senza versione). Se nel documento sono presenti immagini, queste andranno inserite in una sottocartella di nome “Immagini”.

3. **Supporto:** contenente solo le tre cartelle:

- (a) **Dizionari:** contenente il file “*Glossario.csv*” ed il file “*VocabSgad.diz*”. Il primo contiene i termini presenti a glossario ed il secondo contiene i termini del dizionario personalizzato usato in Hunspell<sup>23</sup>.
- (b) **Script:** contenente:
  - i. una cartella per ogni script contenente le sottocartelle “bin” (con i compilati) e “src” (con i sorgenti).
- (c) **Template\_Latex:** contenente solo le due cartelle:
  - i. **Immagini:** contenente le immagini comuni a tutti i documenti (esempio: “*Creazione.png*”).
  - ii. **Template:** contenente i template base per la creazione dei documenti. Conterrà anche la sottocartella “Comandi” in cui verranno inseriti i file tex con i comandi L<sup>A</sup>T<sub>E</sub>X personalizzati.

#### 4.1.3.4.3 Norme sulla commit

Ogni qual volta si termina una modifica ad un file, va eseguita la conferma tramite il comando `commit`. Essa va lanciata specificandone il motivo della commit tramite il seguente comando: `git commit -m “motivo” nomefile`. Il motivo della commit deve indicare il nome del file coinvolto ed il motivo della modifica. Se vengono modificati più file va eseguita una commit per ogni singolo file. È buona norma scrivere una descrizione sintetica e leggibile delle modifiche effettuate ai file che si vuole aggiornare. Tale procedura va eseguita solo alla fine dell’inserimento di tutte le modifiche assegnate. La descrizione deve iniziare obbligatoriamente con la sigla del file inserita tra parentesi quadre. Tale sigla deve essere seguita da uno spazio e dal testo della descrizione (esempio: [SdF] Inserito paragrafo relativo al capitolato 2).

Il diario delle modifiche va aggiornato secondo le regole esposte nella sezione 3.1. Tale operazione va fatta obbligatoriamente prima di eseguire la commit.

### 4.1.4 Strumenti

#### 4.1.4.1 Sistema operativo

Tutti i membri del team operano su un *sistema operativo*<sub>[g]</sub> *Linux*<sub>[g]</sub> distribuzione *Ubuntu*<sub>[g]</sub> x64 nelle versioni maggiori o uguali alla 12.04LTS. Per lo sviluppo di documentazione sono accettati anche *sistemi operativi*<sub>[g]</sub> *Windows*<sub>[g]</sub> 8.1 x64, nelle versioni Home e Professional.

#### 4.1.4.2 Dropbox

Si è deciso di utilizzare *Dropbox*<sub>[g]</sub> per condividere tra i membri del team file e documenti non soggetti a controllo di versione. Tramite tale strumento si condividono anche eventuali manuali utili per la formazione del team.

#### 4.1.4.3 Google Calendar

*Google Calendar*<sub>[g]</sub> viene utilizzato dai membri del team per segnare le date di importanti scadenze, come milestone, incontri o riunioni.

---

<sup>23</sup>Per chiarimenti si veda 3.2.3.2 Hunspell.

#### 4.1.4.4 MyTinyToDo

Questa applicazione web è stata fortemente personalizzata per adattarsi ai bisogni del team. Tale applicazione viene utilizzata per la gestione dei task<sup>24</sup>.

#### 4.1.4.5 Facebook

È stato impostato un gruppo di lavoro privato interno al *social network*<sub>[g]</sub> per gestire le comunicazioni interne<sup>25</sup>.

#### 4.1.4.6 ProjectLibre

Si utilizza *ProjectLibre*<sub>[g]</sub> per il supporto alla pianificazione del progetto e per la gestione delle risorse.

La scelta è dovuta alla completezza del programma e al supporto per i diagrammi *WBS*<sub>[g]</sub>, assente in programmi simili come *GanttProject*<sub>[g]</sub>.

Il software è reperibile su <http://www.projectlibre.org>.

#### 4.1.4.7 GitHub

La scelta di *GitHub*<sub>[g]</sub> per mantenere lo storico di tutte le modifiche apportate ai file e ai documenti relativi al progetto è dovuta alla richiesta del proponente stesso. Il sito GitHub mette a disposizione dello spazio di hosting gratuito, il quale sarà sfruttato dal team per tenere una copia del repository del progetto.

#### 4.1.4.8 Git

Poiché GitHub si basa sul software di versionamento *Git*<sub>[g]</sub>, viene utilizzato unicamente tale strumento.

#### 4.1.4.9 Prezi

Per le presentazioni formali del gruppo, ovvero per le presentazioni che accompagnano ogni revisione di progetto, si è deciso di utilizzare il software *Prezi*<sub>[g]</sub>.

Prezi è uno strumento che permette di creare presentazioni direttamente online, apportando modifiche in modo concorrente. È infatti uno strumento collaborativo dove più persone possono lavorare insieme. Esso mette a disposizione un'interfaccia utente di tipo *ZUI*<sub>[g]</sub>, che permette all'utente di navigare attraverso i contenuti mediante un sistema di zoom.

Il gruppo si ritiene soddisfatto dall'utilizzo di questo software dal punto di vista della resa grafica. Sono invece stati riscontrati alcuni *bug*<sub>[g]</sub> che però non ne hanno compromesso l'utilizzo.

---

<sup>24</sup>Si veda questo stesso documento nella sottosezione 4.1.1.3 Ticketing.

<sup>25</sup>Si veda questo stesso documento nella sottosezione 4.1.1.1.1 Comunicazioni interne.

## A Lista di controllo

Di seguito viene presentata la lista di controllo con gli errori più comuni da controllare durante una inspection:

- norme stilistiche:
  - elenchi puntati e numerati: non terminano con il punto e virgola se è l'ultimo elemento;
  - elenchi puntati e numerati: non terminano con il punto e virgola (a meno dell'ultimo elemento) se la parola che li precede non termina con il punto;
  - nome documento: non viene utilizzata la macro predisposta e non viene indicata la versione del documento dove necessaria;
  - ruoli di progetto: non vengono scritti con la maiuscola iniziale;
  - [ISO/IEC]: scrittura senza inserimento tra parentesi quadre;
  - riferimenti: viene scritto “Vedi la sezione” anziché “Si veda la sezione”, e non nelle note a piè di pagina;
  - punteggiatura e note a piè di pagina: se si trova una nota al termine di una frase, va inserita la macro relativa prima del punto di termine frase;
  - note a piè di pagina: non cominciano con la lettera maiuscola, non terminano con il punto.
- italiano:
  - virgole: vengono utilizzate troppe virgole che spezzano il ritmo di lettura della frase;
  - carattere ‘È’: non viene scritto correttamente utilizzando il comando apposito;
  - caratteri accentati: vengono erroneamente scritti utilizzando l'accento acuto e non l'accento grave;
  - periodi: frasi troppo lunghe rendono i concetti di difficile comprensione;
  - suo/proprio: utilizzo erroneo del termine “suo” dove sostituibile da “proprio”;
  - fase: utilizzo erroneo del termine “fase”;
  - proponente e committente: viene confuso il loro significato.
- UML:
  - attori: il sistema non è ad attore;
  - ortografia: non viene corretta l'ortografia, non automatizzabile trattandosi di immagini;
  - interfacce: non viene utilizzata la notazione di UML 2.0 per la rappresentazione delle interfacce;
  - associazioni:
    - \* non si utilizza la notazione UML 2.0 per la rappresentazione delle associazioni tra classi;

- \* non è utilizzata l'associazione corretta (esempio: associazione scambiata per uso o viceversa);
- \* generalizzazione scambiata per realizzazione e viceversa;
- \* scambiato il verso delle associazioni.
- metodi: non si specificano i tipi di ritorno e dei parametri;
- variabili e metodi: non scritti come specificato alla sezione 2.1.2.4 Nomi e norme stilistiche;
- **L<sup>A</sup>T<sub>E</sub>X**:
  - spaziatura: non viene scritto il comando di spaziatura dopo le macro che lo richiedono;
  - label e caption: non vengono inseriti gli elementi corretti subito dopo l'inserimento di un'immagine o di una tabella;
  - riferimenti: non viene scritto correttamente il riferimento, risultando in una comparsa dei caratteri “??”, o viene utilizzato il tag “hyperref” quando dovrebbero venire utilizzati “ref” e/o “nameref”.
- glossario:
  - segnalazioni: non viene indicata con la macro relativa la prima occorrenza dei termini nel glossario, ad esclusione dei termini sotto la sezione “Riferimenti” e di quelli utilizzati come nomi.
- codice Scala:
  - utilizzo dei punti e virgola quando non necessario;
  - utilizzo di `var` al posto di `val` quando non necessario;
  - utilizzo di collezioni mutabili quando non necessario al posto delle collezioni immutabili;
  - non utilizzo delle `case class` quando invece erano più adatte, visto soprattutto la loro natura immutabile;
  - non rispetto delle norme stilistiche adottate;
  - dimenticare il `private` su metodi e attributi che nella definizione di prodotto erano segnati come privati;
  - mandato messaggi ad attori contenenti classi mutabili non rispettando la convenzione *Akka<sub>[g]</sub>* sui tipi degli oggetti da passare come messaggi.
  - metodi che non producono modifiche sul chiamante dichiarati con le parentesi e viceversa, metodi che modificano lo stato dichiarati con le parentesi;
  - metodo `toString` chiamato senza parentesi.
- codice JavaScript:
  - utilizzo sbagliato delle closure che portano ad una condivisione della memoria;
  - omissione del carattere ‘;’ per terminare uno statement;



- utilizzo dell'operatore '==' piuttosto dell'operatore '===';
- utilizzo dell'operatore '!=' piuttosto dell'operatore '!==';
- omissione della notazione "@override" nel JSDoc dei metodi;
- omissione della notazione "@private" nel JSDoc degli attributi;
- invocazione di metodi fornendo parametri con tipo non corrispondente a quello aspettato.

## B Screenshot MyTinyToDo

**ProTech - Ticketing System** [Grafici Andamento](#) | [Esci](#)

Maurizio Stefano Gallo Gatto Alberto Fabio Glossario Errori + Tutti i tasks ▼

Nuovo task ▶  🔍

**Tasks (20)** ▼ Note: [Mostra](#) / [Nascondi](#) Tags ▼

<input type="checkbox"/>	[NdP] Ristrutturare sezione Organizzazione [Maurizio] [Accettato] Stefano [IndP] [Maurizio] [Accettato] creato il 28 Nov 2013 14:51 → 29 Nov
<input type="checkbox"/>	[NdP] Creazione Diagramma Attività di Ticketing [Maurizio] [Completato] Stefano [IndP] [Maurizio] [Completato] creato il 28 Nov 2013 15:11 → 29 Nov
▶ <input type="checkbox"/>	[Supporto] Realizzare la web app per l'analisi dei requisiti [Maurizio] Gatto [Supporto] [Maurizio] creato il 28 Nov 2013 15:15 → 27 Nov
<input type="checkbox"/>	[PdQ] stendere sezione strumenti [Gallo] [Completato] Alberto [PdQ] [Gallo] [Completato] creato il 28 Nov 2013 15:17 → 25 Nov
<input type="checkbox"/>	[PdQ] stendere sezione disponibili [Gallo] [Completato] Alberto [PdQ] [Gallo] [Completato] creato il 28 Nov 2013 15:17 → 25 Nov
▶ <input type="checkbox"/>	[Supporto] Realizzare la web app per l'analisi dei requisiti [Maurizio] [In attesa] Fabio [Supporto] [Maurizio] [In attesa] creato il 28 Nov 2013 15:18 → 28 Nov
▶ <input type="checkbox"/>	[Supporto] Realizzare codice php per stampare latex di AdR [Maurizio] [Sospeso] Gatto [Supporto] [Maurizio] [Sospeso] creato il 2 Dic 2013 16:59 → da 3 giorni
<input type="checkbox"/>	[SdF] Verificare lo studio di fattibilità [Stefano] [Completato] Gatto [SdF] [Stefano] [Completato] creato il 6 Dic 2013 13:23 → ieri
▶ <input type="checkbox"/>	[Verbal] Redigere il verbale della riunione di giovedì con FunGo [Stefano] [Completato] Gatto [Verbal] [Stefano] [Completato] creato il 6 Dic 2013 13:26 → ieri
▶ <input type="checkbox"/>	[PdP] Correggere errori nella sezione Ciclo di Vita [Stefano] [In attesa] Gallo [PdP] [Stefano] [In attesa] creato il 6 Dic 2013 14:46 → da 3 giorni
<input type="checkbox"/>	[NdP] Sistemare indice Gulpease del documento [Fabio] [Completato] Alberto [IndP] [Fabio] [Completato] creato il 6 Dic 2013 14:56 → da 3 giorni
▶ <input type="checkbox"/>	[PdQ] Correggere errori individuati [Gallo] [Completato] Alberto [PdQ] [Gallo] [Completato] creato il 6 Dic 2013 16:34 → da 2 giorni
▶ <input type="checkbox"/>	[NdP] Correggere errori individuati [Gallo] [Completato] Alberto [IndP] [Gallo] [Completato] creato il 6 Dic 2013 16:36 → da 2 giorni
<input type="checkbox"/>	[AdR] Stendere Use Cases [Stefano] [In corso] Maurizio [AdR] [Stefano] [In corso] creato il 9 Dic 2013 15:08 → domani

Figura 5: Vista generale della pagina iniziale di MyTinyToDo

**ProTech - Ticketing System** [Grafici Andamento](#) | [Esci](#)

Maurizio Stefano Gallo Gatto Alberto Fabio Glossario Errori + Tutti i tasks ▼

Nuovo task ▶  🔍

**Tasks (9)** ▼ Note: [Mostra](#) / [Nascondi](#) Tag: [Stefano] Tags ▼

▶ <input type="checkbox"/>	[PdQ] Il documento non compila. Errore nella linea 336 [Stefano] [Completato] Gallo [PdQ] [Stefano] [Completato] [Stefano] [In corso] creato il 28 Nov 2013 15:12 → da 2 giorni
▶ <input type="checkbox"/>	[Supporto] Lo script Gulpease lascia il file out.txt. Non deve essere lasciato. Vedi ulteriori note [Stefano] [In corso] [Stefano] [In corso] creato il 28 Nov 2013 15:12 → da 2 giorni
▶ <input type="checkbox"/>	[PdQ] Riempire la tabella nel riepilogo delle metriche [Stefano] [Completato] Gallo [PdQ] [Stefano] [Completato] creato il 28 Nov 2013 15:12 → da 2 giorni
<input type="checkbox"/>	[PdP] stendere sezione ciclo di vita [Stefano] [Completato] Gallo [PdP] [Stefano] [Completato] creato il 28 Nov 2013 15:13 → da 2 giorni
▶ <input type="checkbox"/>	[NdP] Correggere errori indicati nella sezione Ticketing [Stefano] Alberto [IndP] [Stefano] creato il 28 Nov 2013 15:15 → oggi
▶ <input type="checkbox"/>	[PdQ] Correggere errori di formattazione nelle sezioni indicate nelle note [Stefano] [Completato] Alberto [PdQ] [Stefano] [Completato] creato il 28 Nov 2013 15:16 → oggi
<input type="checkbox"/>	[NdP] Stendere sezione Lista di controllo [Stefano] Alberto [IndP] [Stefano] creato il 28 Nov 2013 15:16 → oggi
<input type="checkbox"/>	[NdP] Crea immagini per la parte di Documentazione Mytiny [Stefano] [In attesa] Fabio [IndP] [Stefano] [In attesa] creato il 28 Nov 2013 15:18 → oggi
▶ <input type="checkbox"/>	[Supporto] Correggere errori e aggiungere funzionalità [Stefano] [Completato] Fabio [Supporto] [Stefano] [Completato] creato il 28 Nov 2013 15:52 → oggi

Figura 6: Funzione di filtro ticket per tag



[<< Indietro](#)

Nuovo Task

Priorità  $\pm 0$  Scadenza 2/4/14

Task

Aggiungere i tipi agli attributi nei package di Client e Server

Tags

Documento: GCL Verificatore: Stefano Stato:

Nota

GCL

AdR

Gloss

GCL

GUG

LP

NdP

PdP

PdQ

ST

SdF

Verb

MITD

RACheL

Script

DdP

MUG

MUA

3D

Salva Cancell

Figura 7: Form di creazione di un ticket

☐ [SdF] Verificare lo studio di fattibilità [Stefano] [Completato] [SdF], [Stefano], [Completato] creato il 6 Dic 2013 13:23

▼

☐ [Supporto] Realizzare codice php per stampare latex di AdR [Maurizio] [Sospeso] [Supporto], [Maurizio], [Sospeso] creato il 2 Dic 2013 16:59

📄

Criare il file php che ciclicamente stampa la sezione degli UG da inserire in AdR

Completamento

Tag di Argomento

Tag Verificatore

Tag di Stato

Scadenza

Figura 8: Analisi della sintassi di un ticket

## C Screenshot RACheL

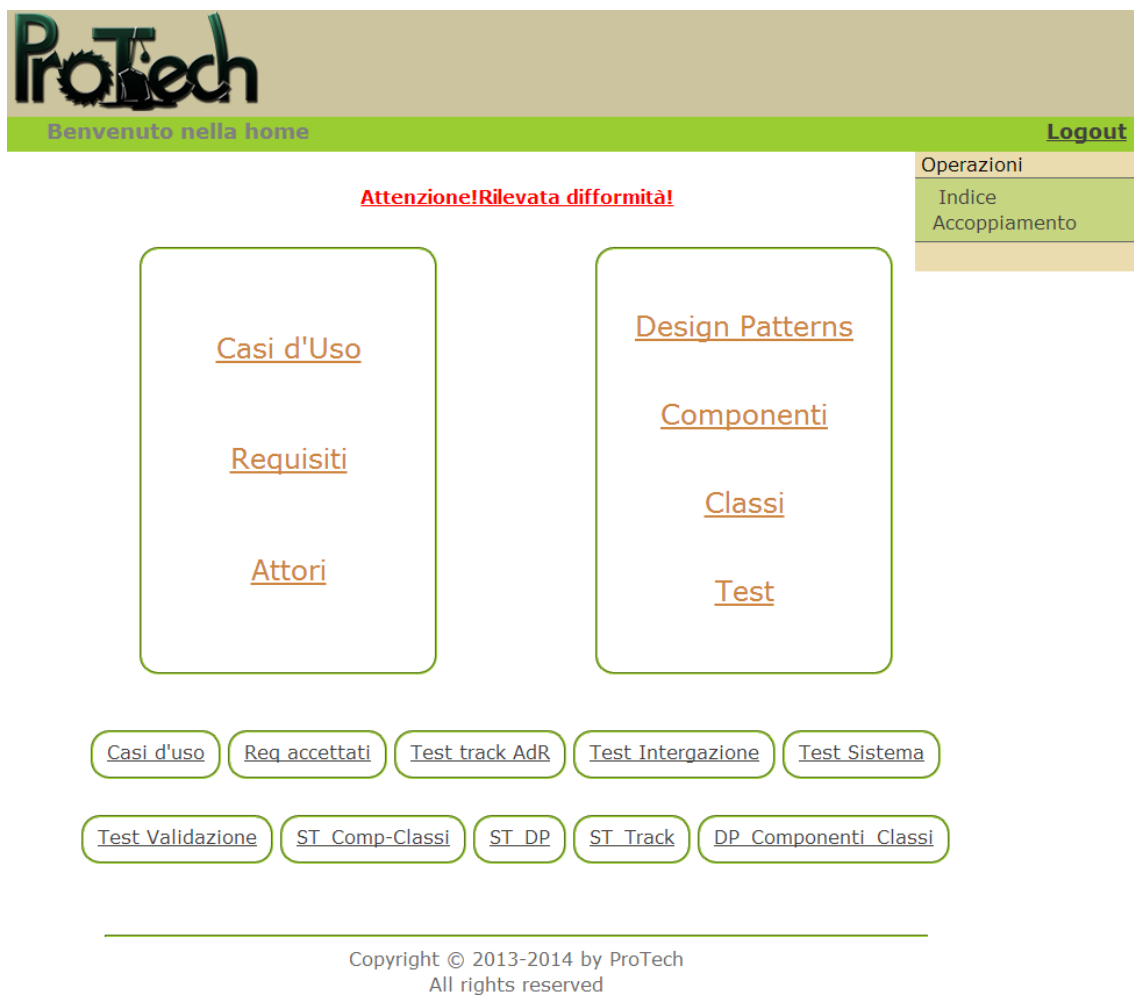


Figura 9: Pagina principale di RACheL con segnalazione errori sui dati


**DP senza classi**

Identificativo
<a href="#">MVC</a>
<a href="#">Three tier</a>

**Requisito padre senza componente di requisito figlio**

Identificativo	Componente
<a href="#">F1</a>	<a href="#">servertier::businesslogic</a>
<a href="#">F1</a>	<a href="#">servertier::presentation</a>

Figura 10: Pagina alert di RACheL




[Home](#)
[Questa pagina traccia tutti casi d'uso](#)
[Logout](#)

[Operazioni](#)
[Inserisci UC](#)

Identificativo	Titolo	Tipologia	
<a href="#">UC 1</a>	Caso d'uso generale	N	<a href="#">Elimina</a>
<a href="#">UC 1.1</a>	Registrazione utente	O	<a href="#">Elimina</a>
<a href="#">UC 1.1.1</a>	Inserimento username	N	<a href="#">Elimina</a>
<a href="#">UC 1.1.2</a>	Inserimento email	N	<a href="#">Elimina</a>
<a href="#">UC 1.1.3</a>	Inserimento password	N	<a href="#">Elimina</a>
<a href="#">UC 1.1.4</a>	Conferma registrazione	N	<a href="#">Elimina</a>
<a href="#">UC 1.2</a>	Autenticazione utente	O	<a href="#">Elimina</a>
<a href="#">UC 1.2.1</a>	Login utente	N	<a href="#">Elimina</a>
<a href="#">UC 1.2.1.1</a>	Inserimento username	N	<a href="#">Elimina</a>
<a href="#">UC 1.2.1.2</a>	Inserimento password	N	<a href="#">Elimina</a>
<a href="#">UC 1.2.1.3</a>	Conferma login	N	<a href="#">Elimina</a>
<a href="#">UC 1.2.2</a>	Autenticazione fallita	E	<a href="#">Elimina</a>

Figura 11: Pagina di ricapitolazione Use Case



[Home](#) » [Use Case](#) » [UC 1](#) » **UC 1.1**









**Operazioni**  
[Inserisci UCA](#)  
[Inserisci UC Figlio](#)

## Visualizza UC: Use Case 1.1

<b>Identificativo</b>	UC 1.1
<b>Titolo</b>	Registrazione utente
<b>Padre</b>	<a href="#">UC 1</a>
<b>Descrizione</b>	Un utente deve poter creare un account per poter eseguire il login all'interno del sistema e usufruire delle funzionalità offerte.
<b>Attori</b>	<a href="#">Utente</a>
<b>Pre-condizione</b>	L'utente accede al sito web del gioco tramite un browser supportato dal sistema.
<b>Post-condizione</b>	Il sistema ha memorizzato i dati relativi all'account utente e ripresenta a questi la schermata iniziale.
<b>Tipologia</b>	O
<b>Numero di Evento</b>	4

## Flusso degli Eventi





### Scenario Principale

id UC	Titolo	Attori	Ordine	
<a href="#">UC 1.1.1</a> 	Inserimento username	<a href="#">Utente</a>	1	
<a href="#">UC 1.1.2</a> 	Inserimento email	<a href="#">Utente</a>	2	
<a href="#">UC 1.1.3</a> 	Inserimento password	<a href="#">Utente</a>	3	
<a href="#">UC 1.1.4</a> 	Conferma registrazione	<a href="#">Utente</a>	4	


### Scenari Alternativi

id UCA	Descrizione	
UCA 1.1.1	L'utente arresta l'inserimento dei dati senza confermare la registrazione di un nuovo account. In tal caso, verrà ricondotto alla schermata .	<a href="#">Elimina</a>

## UC Figli

id UC	Titolo	Tipo
<a href="#">UC 1.1.1</a> 	Inserimento username	N
<a href="#">UC 1.1.2</a> 	Inserimento email	N
<a href="#">UC 1.1.3</a> 	Inserimento password	N
<a href="#">UC 1.1.4</a> 	Conferma registrazione	N

## Requisiti

id Req	Descrizione
R0F1 	Un utente può creare un account per usufruire delle funzionalità offerte.

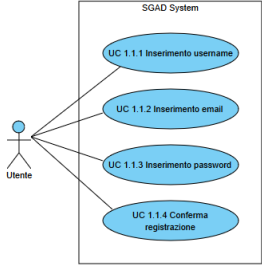



Figura 12: Pagina per Use Case



Home » Use Case » Crea UC

**Crea UC**

Identificativo UC	<input type="text"/>
Titolo	<input type="text"/>
Padre	1 <input type="button" value="v"/>
Descrizione	<div></div>
Pre-condizione	<div></div>
Post-condizione	<div></div>
Tipologia	N <input type="button" value="v"/>
Numero di Evento	<input type="text"/>

Salva

Figura 13: Pagina di inserimento Use Case



[Home](#)
[Questa pagina traccia tutti i requisiti](#)
[Logout](#)

[Operazioni](#)
[Insert requirement](#)

## RF

Identificativo	Descrizione	
R0F1	Un utente deve poter creare un account per poter eseguire il login all'interno del sistema e usufruire delle funzionalità offerte.	<a href="#">Elimina</a>
R0F1.1	Il modulo di registrazione richiede una username all'utente che vuole registrarsi.	<a href="#">Elimina</a>
R0F1.1.1	Lo username deve essere univoco in tutto il sistema.	<a href="#">Elimina</a>
R0F1.2	Il modulo di registrazione richiede l'email dell'utente che vuole autenticarsi.	<a href="#">Elimina</a>
R0F1.2.1	La email deve avere un formato valido. Di conseguenza deve contenere il carattere \sq{@} preceduto da altri caratteri e seguito da un dominio.	<a href="#">Elimina</a>
R0F1.3	Il modulo di registrazione richiede una password all'utente che vuole registrarsi.	<a href="#">Elimina</a>
R0F1.3.1	La password inserita deve essere di una lunghezza almeno di 8 e al massimo di 16 caratteri. La password deve contenere almeno un carattere e almeno un numero.	<a href="#">Elimina</a>
R0F1.4	Alla creazione dell'account viene creato anche il villaggio dell'utente.	<a href="#">Elimina</a>
R2F7.3.1	La rimozione richiede la selezione di un server non appartenente al cluster.	<a href="#">Elimina</a>
R2F8	Il sistema deve essere aggiornato senza provocare un downtime generale del server.	<a href="#">Elimina</a>

## RP

Identificativo	Descrizione	
----------------	-------------	--

## RQ


Identificativo	Descrizione	
R0Q1	Viene fornito il manuale utente giocatore.	<a href="#">Elimina</a>
R0Q2	Devono essere rispettate tutte le norme e le metriche sulla stesura di codice riportate nei documenti \file{Norme di Progetto v1.00} e \file{Piano di Qualifica v1.00}.	<a href="#">Elimina</a>
R0Q3	Dovranno essere rispettate tutte le norme di progettazione architeturale e di dettaglio indicate nel documento \file{Norme di Progetto v1.00}. Per la progettazione di dettaglio dovranno essere rispettate anche le metriche indicate in \file{Piano di Qualifica v1.00}.	<a href="#">Elimina</a>
R2Q4	Viene fornito il manuale utente amministratore.	<a href="#">Elimina</a>

## RV

Identificativo	Descrizione	
R0V1	Il dati devono essere salvati su un database di tipo documentale. È implementato con Mongo o Couchbase.	<a href="#">Elimina</a>
R0V2	L'architettura server funziona distribuita in un cluster di server.	<a href="#">Elimina</a>
R0V3	È fornito il documento \file{Report sui test di performance}.	<a href="#">Elimina</a>
R0V4	È utilizzato uno dei seguenti linguaggi per l'implementazione delle API e degli attori: \begin{itemize} \item Ruby insieme al framework Dcell; \item Scala insieme al framework Akka. \end{itemize}	<a href="#">Elimina</a>
R0V5	Il progetto deve essere pubblicato su GitHub.	<a href="#">Elimina</a>
R0V5.1	Si devono impiegare le issue di GitHub per segnalare i bug.	<a href="#">Elimina</a>
R0V6	Il sistema lato client deve essere compatibile con almeno uno dei seguenti browser: \begin{itemize} \item \textbf{Firefox}: versione 23.0 o superiore \item \textbf{Chrome}: versione 29.0.1597.76 o superiore. \end{itemize}	<a href="#">Elimina</a>
R0V6.1	La parte di prodotto relativa all'architettura lato client deve funzionare su Windows 7 o superiore.	<a href="#">Elimina</a>
R0V7	La parte di prodotto relativa all'architettura lato server deve funzionare su Ubuntu 12.04LTS x64 o superiore.	<a href="#">Elimina</a>
R0V8	Il sistema deve essere progettato e programmato secondo il paradigma ad attori.	<a href="#">Elimina</a>

Figura 14: Pagina ricapitolazione requisiti










[Home](#) » [Requisiti](#) » **R0F1**

**Visualizza Requisiti: R0F1**

<b>Identificativo</b>	R0F1
<b>Tipo Requisito</b>	Funzionale
<b>Padre</b>	null
<b>Importanza</b>	0
<b>Descrizione</b>	Un utente può creare un account per usufruire delle funzionalità offerte.
<b>Impl</b>	No
<b>Da Rispettare</b>	Si

<b>Operazioni</b>	
<a href="#">Aggiungi Fonte</a>	
<a href="#">Inserisci Req Figlio</a>	

**Requisiti Figli**

Identificativo	Descrizione
R0F1.1 	La registrazione richiede lo username dell'utente.
R0F1.2 	La registrazione richiede l'email dell'utente.
R0F1.3 	La registrazione richiede una password.
R0F1.4 	Alla creazione dell'account viene creato anche il villaggio dell'utente.
R0F1.5 	Il sistema comunica all'utente gli eventuali errori sui dati inseriti durante la procedura di registrazione.

**Fonti del Requisito**

Fonte	Operazione
UC 1.1	Elimina
Capitolato	Elimina

**Componenti**

Componente
<a href="#">clienttier::controller</a>
<a href="#">clienttier::view</a>
<a href="#">servvertier</a>

Figura 15: Pagina relativa ad un requisito



Figura 16: Pagina di inserimento requisiti



Identificativo	Nome	Elimina
1	Utente	<a href="#">Elimina</a>
2	Utente giocatore autenticato	<a href="#">Elimina</a>
3	Utente amministratore autenticato	<a href="#">Elimina</a>
4	Utente autenticato	<a href="#">Elimina</a>

Figura 17: Pagina di riepilogo attori

[Home](#) » [Attori](#) » **Attore Utente**

## Visualizza Attore: Utente

<b>Identificativo</b>	Att 1
<b>Nome</b>	Utente

## Responsabilità

### UC

id UC	Titolo
<a href="#">UC 1</a>	Caso d'uso generale
<a href="#">UC 1.1</a>	Registrazione utente
<a href="#">UC 1.1.1</a>	Inserimento username
<a href="#">UC 1.1.2</a>	Inserimento email
<a href="#">UC 1.1.3</a>	Inserimento password
<a href="#">UC 1.1.4</a>	Conferma registrazione
<a href="#">UC 1.2</a>	Autenticazione utente
<a href="#">UC 1.2.1</a>	Login utente
<a href="#">UC 1.2.1.1</a>	Inserimento username
<a href="#">UC 1.2.1.2</a>	Inserimento password
<a href="#">UC 1.2.1.3</a>	Conferma login

Figura 18: Pagina relativa ad un attore

## Visualizza:Strategy

<b>Identificativo</b>	Strategy
<b>Scopo</b>	Il pattern Strategy definisce una famiglia di algoritmi, incapsulati e resi intercambiabili. Strategy permette agli algoritmi di variare indipendentemente dai client che ne fanno uso.
<b>Tipo</b>	Design pattern comportamentali
<b>Contesto</b>	È usato dalla logica di controllo del server per eseguire operazioni diverse in base al tipo di richiesta del client. Essendo l'operazione un algoritmo diverso per analizzare i dati, il pattern risulta adatto allo scopo. Ogni operazione non mantiene uno stato interno che varia per ogni utente quindi non risulta necessario istanziare un oggetto per effettuare l'operazione per ogni utente. Il pattern viene utilizzato assieme al pattern Flyweight per mantenere un unico oggetto per ogni operazione. Ciò comporta significativi risparmi di spazio.

## Classi

Classe	
<a href="#">sgad::servtier::businesslogic::operations::AttackUserVillage</a>	<a href="#">Elimina</a>
<a href="#">sgad::servtier::businesslogic::operations::BuildConstruction</a>	<a href="#">Elimina</a>
<a href="#">sgad::servtier::businesslogic::operations::DemolishBuilding</a>	<a href="#">Elimina</a>
<a href="#">sgad::servtier::businesslogic::operations::DismissUnit</a>	<a href="#">Elimina</a>
<a href="#">sgad::servtier::businesslogic::operations::DonateResources</a>	<a href="#">Elimina</a>
<a href="#">sgad::servtier::businesslogic::operations::DonateUnits</a>	<a href="#">Elimina</a>
<a href="#">sgad::servtier::businesslogic::operations::GetAllServerData</a>	<a href="#">Elimina</a>
<a href="#">sgad::servtier::businesslogic::operations::GetServerData</a>	<a href="#">Elimina</a>
<a href="#">sgad::servtier::businesslogic::operations::HarvestResource</a>	<a href="#">Elimina</a>
<a href="#">sgad::servtier::businesslogic::operations::LoadFinishedBuilding</a>	<a href="#">Elimina</a>
<a href="#">sgad::servtier::businesslogic::operations::LoadGlobalData</a>	<a href="#">Elimina</a>
<a href="#">sgad::servtier::businesslogic::operations::LoadVillage</a>	<a href="#">Elimina</a>
<a href="#">sgad::servtier::businesslogic::operations::LoginCheck</a>	<a href="#">Elimina</a>
<a href="#">sgad::servtier::businesslogic::operations::Logout</a>	<a href="#">Elimina</a>
<a href="#">sgad::servtier::businesslogic::operations::Operation</a>	<a href="#">Elimina</a>
<a href="#">sgad::servtier::businesslogic::operations::Registration</a>	<a href="#">Elimina</a>
<a href="#">sgad::servtier::businesslogic::operations::StealResources</a>	<a href="#">Elimina</a>
<a href="#">sgad::servtier::businesslogic::operations::TrainUnit</a>	<a href="#">Elimina</a>
<a href="#">sgad::servtier::businesslogic::operations::UpgradeBuilding</a>	<a href="#">Elimina</a>

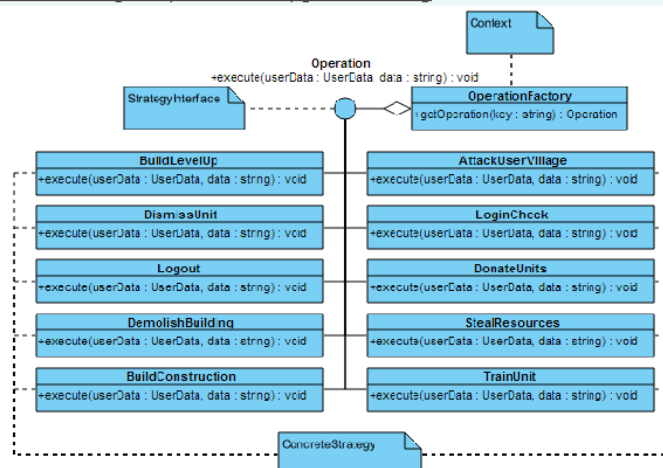



Figura 19: Pagina relativa ad un design pattern



[Home](#) » [Componenti](#) » [sgad::clienttier::view](#)

### Visualizza: view

Identificativo	sgad::clienttier::view
Descrizione	Componente per la componente View dell'architettura MVC.
SuperPackage	<a href="#">sgad::clienttier</a>
Da Stampare	Si

Operazioni

[Inserisci Figlia](#)

[Inserisci Classe](#)

[Inserisci Relazione](#)

[Inserisci Requisito](#)

### Classi Contenute

Classe
Nessuna Classe

### Package Contenuti

Package
<a href="#">sgad::clienttier::view::commands</a>
<a href="#">sgad::clienttier::view::context</a>
<a href="#">sgad::clienttier::view::graphicobjects</a>

### Relazioni con Componenti

Componente	
<a href="#">sgad::clienttier::controller</a>	<a href="#">Elimina</a>
<a href="#">sgad::clienttier::model</a>	<a href="#">Elimina</a>

### Requisiti

Requisito	Descrizione	
<a href="#">R0F5.1.1.1</a>	La raccolta di risorse richiede la selezione di un edificio produttivo.	<a href="#">Elimina</a>
<a href="#">R0F5.1.4.2</a>	La costruzione richiede la selezione di una casella libera dove posizionare il nuovo edificio.	<a href="#">Elimina</a>
<a href="#">R2F5.2.1.4.2</a>	È richiesta la selezione dell'edificio da cui prelevare le risorse.	<a href="#">Elimina</a>
<a href="#">R1F5.2.2.2</a>	Il dono richiede la scelta del tipo di risorsa da donare.	<a href="#">Elimina</a>
<a href="#">R0F6</a>	L'utente giocatore autenticato deve potersi deautenticare dal sistema.	<a href="#">Elimina</a>

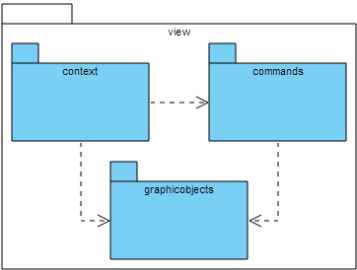



Figura 20: Pagina relativa ad un componente



[Home](#) » [Classi](#) » [sgad::clienttier::controller::actions::DemolishBuilding](#)

## Visualizza:DemolishBuilding

<b>Identificativo</b>	sgad::clienttier::controller::actions::DemolishBuilding
<b>Descrizione</b>	Classe per la gestione della demolizione di un edificio già presente nel villaggio.
<b>Package</b>	sgad::clienttier::controller::actions
<b>Utilizzo</b>	Viene utilizzata per permettere all'utente di demolire un edificio che ha precedentemente costruito.
<b>Da Stampare</b>	Si

**Eredita da**

Classe	
<a href="#">sgad::clienttier::controller::actions::Action</a>	<a href="#">Elimina</a>

**Metodi**

Tipo di Ritorno	Metodo	
Costruttore	<a href="#">DemolishBuilding(building : BuildingPossession)</a>	<a href="#">Elimina</a>

**Attributi**

Attributo
Nessun Attributo

**Viene Ereditata da**

Classe
Nessun Erede

**Relazioni con Classi**

Classe	
<a href="#">sgad::clienttier::model::personaldata::BuildingPossession</a>	<a href="#">Elimina</a>

**Design Pattern**

Design	
Composite	<a href="#">Elimina</a>

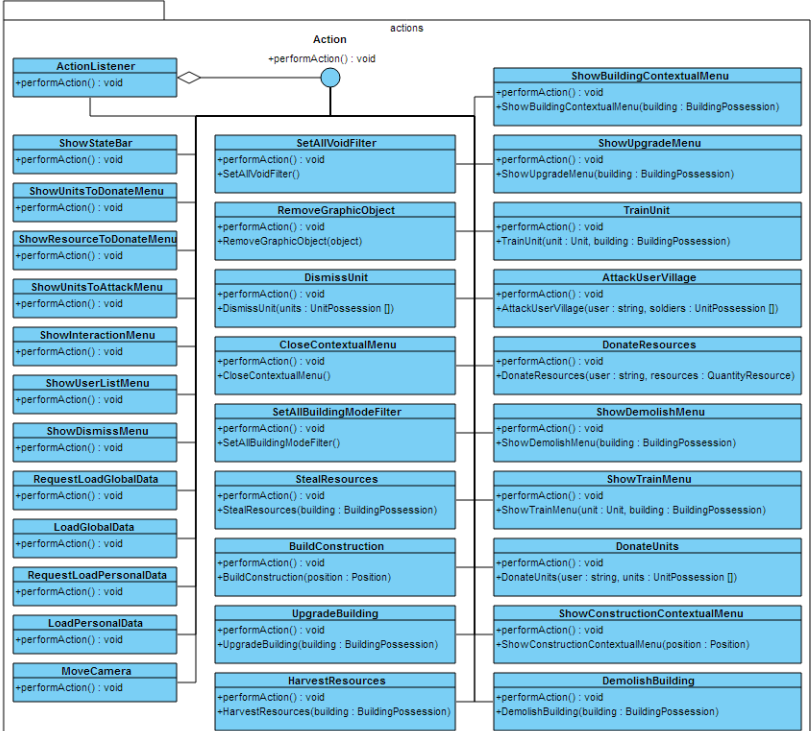


Figura 21: Pagina relativa ad una classe

## D Screenshot Tracker

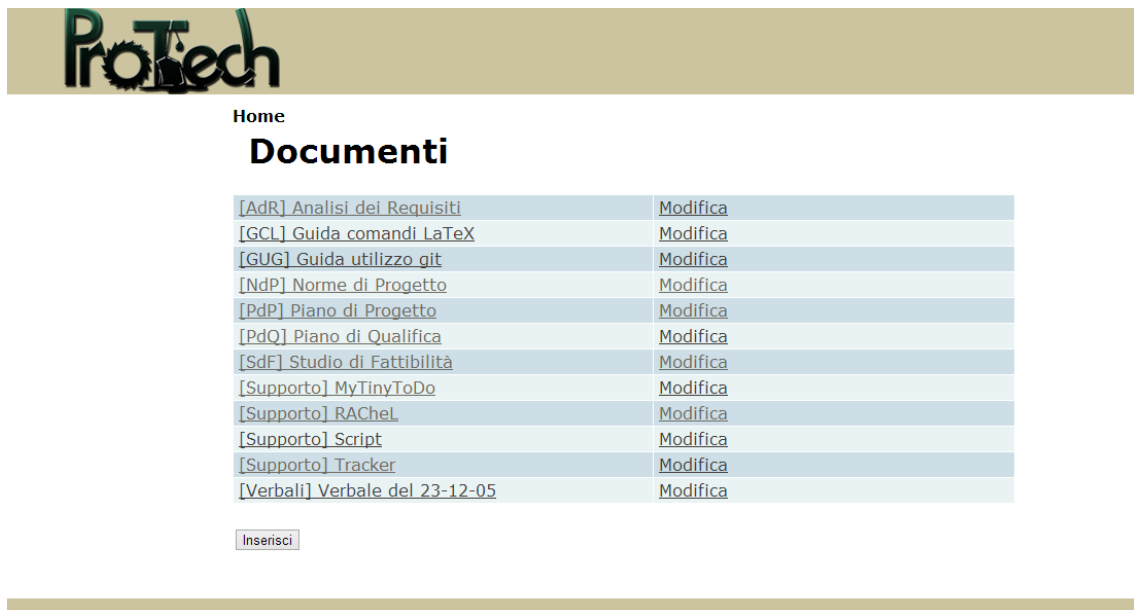


Figura 22: Pagina principale di Tracker

[Home](#) » **[NdP] Norme di Progetto**

## **[NdP] Norme di Progetto**

<input type="checkbox"/>	[Struttura]	[10,7] [Strumenti di verifica del software] Ne si parla in futuro? Lo strumento di validazione W3C va in una categoria a se? Scegliere quale usare.	<a href="#">Modifica</a>
<input type="checkbox"/>	[Sintattico]	7.5 l'esempio di motivo è praticamente incomprensibile	<a href="#">Modifica</a>
<input type="checkbox"/>	[Validazione]	8.1 [MTTD] non è tra virgolette	<a href="#">Modifica</a>
<input type="checkbox"/>	[Validazione]	8.4 Inserire riferimenti a Tracker	<a href="#">Modifica</a>
<input type="checkbox"/>	[Validazione]	9.3.1 Il repository non sta più su GitHub	<a href="#">Modifica</a>
<input type="checkbox"/>	[Validazione]	9.4 Paragrafi rientranti del doppio	<a href="#">Modifica</a>
<input type="checkbox"/>	[Contenuto]	9.9 calcolo gulpease : viene generato il file Gulpease.txt	<a href="#">Modifica</a>
<input type="checkbox"/>	[Validazione]	Bisogna rivedere la validazione dei riferimenti infra documento perchè	<a href="#">Modifica</a>

Figura 23: Pagina relativa agli errori di un documento