

SiVoDiM

Sintesi Vocale per Dispositivi Mobili



Norme di Progetto

Versione	3.0.0
Redattori	Enrico Chiara Gino Zaidan Riccardo Rizzo
Verificatori	Alberto Andriolo
Responsabili	Federico Rossetto
Uso	Interno
Lista di distribuzione	Stark Labs Prof. Tullio Vardanega, Prof. Riccardo Cardin

Documento contenente l'insieme di norme rispettate dal gruppo Stark Labs per la realizzazione del progetto SiVoDiM

Registro delle modifiche

Versione	Data	Attività	Autori
3.0.0	30/05/2016	Accettazione	Federico Rossetto
2.1.0	30/05/2016	Revisione delle modifiche apportate	Alberto Andriolo
2.0.7	27/05/2016	Sezione 3.2.2: Aggiunta la sezione 3.2.2.3 Strumenti	Enrico Chiara
2.0.6	27/05/2016	Eliminate eccessive ripetizioni del termine "processo"	Riccardo Rizzo
2.0.5	27/05/2016	Sezione 3.4: Completata la sezione 3.4 Verifica	Riccardo Rizzo
2.0.4	27/05/2016	Sezioni 3.3 e 3.4: Separata Validazione da Verifica	Riccardo Rizzo
2.0.3	24/05/2016	Sezione 3.2.2: Aggiunta la sezione 3.2.2.2 Procedure	Enrico Chiara
2.0.2	24/05/2016	Sezione 2.2.2.3: Ampliata la sezione Codifica dei file	Riccardo Rizzo
2.0.1	24/05/2016	Sezione 4.2.2.4: Aggiunta la sezione Android Studio	Riccardo Rizzo
2.0.0	05/05/2016	Accettazione	Riccardo Rizzo
1.1.0	04/05/2016	Verifica	Federico Rossetto
1.0.3	04/05/2016	Sezione 2.2.2.7 a 2.2.2.13: Aggiunte le sezioni riguardanti la Progettazione	Enrico Chiara
1.0.2	04/05/2016	Sezione 2.2.1.2 - 2.2.1.3: Aggiunte le sezioni Progettazione architettonale e Progettazione di dettaglio	Enrico Chiara
1.0.1	29/04/2016	Sezione 3.2.1 - 3.2.2 - 3.2.3: Aggiunte le sezioni Test, Metriche e Obiettivi inerenti la Qualità	Enrico Chiara
1.0.0	11/03/2016	Accettazione	Alberto Andriolo
0.3.0	10/03/2016	Verifica	Federico Rossetto
0.2.1	10/03/2016	Correzione errori	Enrico Chiara
0.2.0	10/03/2016	Verifica	Riccardo Rizzo
0.1.1	09/03/2016	Correzione errori	Enrico Chiara

0.1.0	09/03/2016	Verifica	Riccardo Rizzo
0.0.6	09/03/2016	Ampliamento Processi Organizzativi	Gino Zaidan
0.0.5	08/03/2016	Stesura Processi Organizzativi	Gino Zaidan
0.0.4	07/03/2016	Ampliamento Processi di Supporto	Riccardo Rizzo
0.0.3	05/03/2016	Stesura Processi di Supporto	Riccardo Rizzo
0.0.2	04/03/2016	Stesura Processi Primari	Enrico Chiara
0.0.1	04/03/2016	Stesura struttura documento	Enrico Chiara

Indice

1 Introduzione	1
1.1 Scopo del documento	1
1.2 Scopo del progetto	1
1.3 Glossario	1
1.4 Riferimenti	2
1.4.1 Informativi	2
2 Processi primari	3
2.1 Fornitura	3
2.1.1 Attività	3
2.1.1.1 Accettazione	3
2.1.1.1.1 Discussione e scelta del capitolato	3
2.1.1.1.2 Studio di fattibilità	3
2.1.1.2 Preparazione della risposta	3
2.1.1.2.1 Definizione e preparazione della proposta	3
2.1.1.3 Pianificazione	3
2.1.1.3.1 Scelta del modello di ciclo di vita	4
2.1.1.3.2 Sviluppo e documentazione del Piano di Progetto	4
2.2 Sviluppo	4
2.2.1 Attività	4
2.2.1.1 Analisi dei requisiti	4
2.2.1.2 Progettazione architettonale	4
2.2.1.3 Progettazione di dettaglio	5
2.2.2 Norme	5
2.2.2.1 Classificazione dei requisiti	5
2.2.2.2 Classificazione dei casi d'uso	6
2.2.2.3 Codifica dei file	6
2.2.2.3.1 Nomi e norme di codifica	6
2.2.2.3.2 Versionamento	7
2.2.2.3.3 Ricorsione	7
2.2.2.4 Inserimento dei requisiti	7
2.2.2.5 Design Pattern	9
2.2.2.6 Diagrammi UML	9
2.2.2.7 Classificazione dei componenti	9
2.2.2.8 Classificazione delle classi	12
2.2.2.9 Descrizione di una classe	12
2.2.2.10 Test di unità	12
2.2.2.11 Test di integrazione	12
3 Processi di supporto	13
3.1 Documentazione	13
3.1.1 Attività	13
3.1.1.1 Documentazione	13
3.1.2 Procedure	13
3.1.2.1 Gestione dei documenti	13
3.1.2.2 Creazione di un nuovo documento	13
3.1.2.3 Avanzamento di un documento	13

3.1.3	Gestione del glossario	15
3.1.4	Norme	15
3.1.4.1	Progettazione e sviluppo dei documenti	15
3.1.4.2	Versionamento	15
3.1.4.3	Template	16
3.1.4.4	Struttura dei documenti	16
3.1.4.4.1	Prima pagina	16
3.1.4.4.2	Registro delle modifiche	17
3.1.4.4.3	Indici	17
3.1.4.4.4	Formattazione di una pagina	17
3.1.4.5	Suddivisione dei documenti	17
3.1.4.5.1	Norme di Progetto	17
3.1.4.5.2	Studio di Fattibilità	17
3.1.4.5.3	Analisi dei Requisiti	17
3.1.4.5.4	Piano di Progetto	18
3.1.4.5.5	Piano di Qualifica	18
3.1.4.5.6	Specifica Tecnica	18
3.1.4.5.7	Definizione di Prodotto	18
3.1.4.5.8	Glossario	18
3.1.4.6	Norme tipografiche	18
3.1.4.6.1	Stile del testo	18
3.1.4.6.2	Punteggiatura	19
3.1.4.6.3	Composizione del testo	19
3.1.4.6.4	Formati	19
3.1.4.6.5	Sigle	19
3.1.4.7	Componenti grafiche	19
3.1.4.7.1	Immagini e diagrammi	20
3.2	Qualità	20
3.2.1	Test	20
3.2.1.1	Classificazione dei test	20
3.2.2	Metriche	20
3.2.2.1	Classificazione delle metriche	20
3.2.2.2	Procedure	21
3.2.2.2.1	Calcolo delle metriche attraverso Android Studio .	21
3.2.2.3	Strumenti	21
3.2.2.3.1	Copertura dei requisiti obbligatori	21
3.2.2.3.2	Copertura dei requisiti desiderabili	21
3.2.2.3.3	Numero livelli di annidamento	21
3.2.2.3.4	Numero di attributi per classe	21
3.2.2.3.5	Numero di parametri per metodo	21
3.2.2.3.6	Linee di codice per commento	21
3.2.2.3.7	Copertura di codice	22
3.2.2.3.8	Percentuale di test superati	22
3.2.2.3.9	Failure avoidance	22
3.2.2.3.10	Breackdown avoidance	22
3.2.3	Obiettivi	22
3.2.3.1	Classificazione degli obiettivi	22
3.3	Verifica	22
3.3.1	Attività	23

3.3.1.1	Analisi statica	23
3.3.1.2	Analisi dinamica	23
3.3.1.3	Gestione anomalie	23
3.3.1.4	Tracciamento	23
3.3.2	Procedure	23
3.3.2.1	Procedure di assegnazione delle anomalie	23
3.3.3	Strumenti	25
3.3.3.1	Correzione ortografica	25
3.3.3.2	Calcolo dell'indice Gulpease	25
3.4	Validazione	25
3.4.1	Responsabilità	25
3.5	Procedure	25
3.5.1	Procedura per la validazione	25
4	Processi organizzativi	26
4.1	Gestione dei processi	26
4.1.1	Attività	26
4.1.1.1	Gestione delle comunicazioni	26
4.1.1.1.1	Comunicazione interna	26
4.1.1.1.2	Comunicazione esterna	26
4.1.1.1.3	Composizione email	26
4.1.1.2	Gestione delle riunioni	27
4.1.1.2.1	Riunioni interne	27
4.1.1.2.2	Riunioni esterne	27
4.1.1.3	Gestione del sistema dei task	28
4.1.1.4	Gestione delle milestone	28
4.1.1.5	Gestione dei task	28
4.1.1.6	Gestione dello svolgimento dei task	29
4.1.2	Procedure	29
4.1.2.1	Generazione di una milestone	29
4.1.2.2	Assegnazione di un task	29
4.1.2.3	Svolgimento di un task	29
4.1.2.4	Rilevamento dei rischi	31
4.1.3	Norme	31
4.1.3.1	Ruoli di Progetto	31
4.1.3.1.1	Responsabile di Progetto	31
4.1.3.1.2	Amministratore di Progetto	33
4.1.3.1.3	Analista	33
4.1.3.1.4	Progettista	33
4.1.3.1.5	Programmatore	33
4.1.3.1.6	Verificatore	33
4.1.4	Strumenti	33
4.1.4.1	TeXstudio	33
4.1.4.2	Teamwork	33
4.1.4.3	Astah	34
4.1.4.4	Microsoft Project 2016	34
4.1.4.5	Telegram	34
4.1.4.6	Microsoft Office PowerPoint	34
4.2	Gestione delle infrastrutture	34

4.2.0.7	Attività	34
4.2.0.7.1	Gestione del repository	34
4.2.0.7.2	Gestione del messaggio di commit	34
4.2.0.8	Procedure	34
4.2.0.8.1	Installazione di Git	34
4.2.0.8.2	Creazione di una cartella locale di repository	35
4.2.0.8.3	Creazione del branch personale	35
4.2.1	Norme	35
4.2.1.1	Repository	35
4.2.1.1.1	Nomi dei file in SiVoDiM	35
4.2.1.1.2	Struttura di SiVoDiM	36
4.2.1.1.3	Messaggio di commit	36
4.2.2	Strumenti	36
4.2.2.1	Git	36
4.2.2.2	GitHub	36
4.2.2.3	Dropbox	36
4.2.2.4	Android Studio	36
4.2.2.5	Sistema Operativo	37
4.3	Formazione dei membri del gruppo	37

Elenco delle figure

1	Diagramma di attività - Inserimento di un caso d'uso	10
2	Diagramma di attività - Inserimento di un requisito	11
3	Diagramma di attività - Avanzamento di un documento	14
4	Diagramma di attività - Gestione del Glossario	15
5	Diagramma di attività - Procedura di assegnazione delle anomalie	24
6	Diagramma di attività - Procedura di assegnazione di un task	30
7	Diagramma di attività - Procedura di svolgimento di un task	32

1 Introduzione

1.1 Scopo del documento

Questo documento definisce le norme che i membri del gruppo Stark Labs si impegnano a rispettare per un corretto svolgimento del progetto SiVoDiM: Sintesi Vocale per Dispositivi Mobili. Ogni componente del gruppo è tenuta a leggere tale documento e seguire le norme con lo scopo di raggiungere il miglior punto di incontro tra efficienza ed efficacia. In questo modo viene garantita l'uniformità del materiale prodotto e vengono facilitate le operazioni di verifica. In particolare, verranno specificate le seguenti norme:

- Interazioni tra i membri del gruppo;
- Comunicazione verso l'esterno;
- Stesura dei documenti e convenzioni tipografiche;
- Organizzazione dell'ambiente di lavoro;
- Modalità di lavoro durante le fasi del progetto;
- Stesura del codice.

1.2 Scopo del progetto

Lo scopo del progetto risiede nello sviluppo di un'applicazione utile a dimostrare efficacemente le potenzialità del motore di sintesi vocale FA-TTS_G, realizzato dall'azienda MIVOQ s.r.l. e messo a disposizione del gruppo di lavoro. Si devono realizzare due applicazioni per sistemi Android_G:

- **Applicazione di configurazione:** deve permettere all'utente di interfacciarsi direttamente con il sistema operativo per configurare, salvare e modificare le voci ereditate dal motore di sintesi FA-TTS di MIVOQ;
- **Applicazione per la creazione di sceneggiati:** permette la creazione e il salvataggio di racconti e sceneggiati, che possono essere esportati in formato audio attraverso l'utilizzo del motore FA-TTS.

Entrambe le applicazioni devono interfacciarsi con un modulo di basso livello:

- **Modulo di sistema:** permette di interfacciarsi tramite connessione di rete al motore FA-TTS. Fornisce una libreria contenente tutte le funzionalità offerte da FA-TTS, utile nell'ottica di un riuso futuro del *software*.

Lo sviluppo di tutte e quattro le suddette componenti è a carico del gruppo Stark Labs.

1.3 Glossario

Al fine di aumentare la comprensione del testo ed evitare eventuali ambiguità, viene fornito un glossario (*Glossario v2.0.0*) contenente le definizioni degli acronimi e dei termini tecnici utilizzati nei documenti. Ogni vocabolo che ha un riferimento contenuto nel glossario è contrassegnato dal pedice “_G”.

1.4 Riferimenti

1.4.1 Informativi

- *Glossario v2.0.0;*
- Capitolato C1 – Actorbase: a NoSQL DB based on the Actor model
<http://www.math.unipd.it/~tullio/IS-1/2015/Progetto/C1.pdf>;
- Capitolato C2 – CLIPS: Communication & Localization with Indoor Positioning Systems
<http://www.math.unipd.it/~tullio/IS-1/2015/Progetto/C2.pdf>;
- Capitolato C3 – UMAP: un motore per l’analisi predittiva in ambiente Internet of Things
<http://www.math.unipd.it/~tullio/IS-1/2015/Progetto/C3.pdf>;
- Capitolato C4 – MaaS: MongoDB as an admin Service
<http://www.math.unipd.it/~tullio/IS-1/2015/Progetto/C4.pdf>;
- Capitolato C5 – Quizzipedia: software per la gestione di questionari
<http://www.math.unipd.it/~tullio/IS-1/2015/Progetto/C5.pdf>;
- Capitolato C6 – SiVoDiM: Sintesi Vocale per Dispositivi Mobili
<http://www.math.unipd.it/~tullio/IS-1/2015/Progetto/C6.pdf>;
- AS/NZS ISO/IEC 12207:1997
http://www.math.unipd.it/~tullio/IS-1/2009/Approfondimenti/ISO_12207-1995.pdf;
- SI/ISO 31-0 https://en.wikipedia.org/wiki/ISO_31-0;
- Guida all’utilizzo di Teamwork_G
<http://support.teamwork.com/projects/start/getting-started>;
- Guida all’utilizzo di GitHub_G
<https://guides.github.com>;
- Guida all’utilizzo di Astah_G
<http://astah.net/tutorials>;
- Guida all’utilizzo di Microsoft Project 2016_G
https://youtu.be/_eD2u8bxecs
<https://support.office.com/it-it/article/Formattazione-del-diagramma-di-una-visualizzazione-di-Gantt-7473acdc-4abe-4b2f-8361-546efa9dce06#top>.

2 Processi primari

In questa sezione vengono descritti la fornitura e lo sviluppo messi in atto dal gruppo di lavoro Stark Labs. L'acquisizione spetta al Committente e al Proponente del capitolato scelto, mentre la manutenzione non può essere eseguita per vincoli dati dal tempo disponibile. Ad ogni modo, il prodotto *software* e la documentazione fornita devono garantire la possibilità futura di essere sottoposti alle attività previste dal processo di manutenzione.

2.1 Fornitura

2.1.1 Attività

2.1.1.1 Accettazione

2.1.1.1.1 Discussione e scelta del capitolato Il *Responsabile di Progetto* ha il compito di organizzare gli incontri per permettere ai componenti del gruppo di discutere sui capitolati disponibili. Le valutazioni che hanno portato a prendere una decisione finale sono documentate nello *Studio di Fattibilità v1.0.0*.

2.1.1.1.2 Studio di fattibilità Per realizzare il documento devono essere presi in considerazione i seguenti punti, adattati ai capitolati disponibili:

- Valutazione generale del capitolato;
- Valutazione dei fattori di rischio.

Per il capitolato scelto devono essere analizzati anche i seguenti punti:

- Studio del dominio applicativo;
- Studio del dominio tecnologico;
- Analisi di mercato;
- Analisi delle potenziali criticità.

2.1.1.2 Preparazione della risposta

2.1.1.2.1 Definizione e preparazione della proposta I membri del gruppo Stark Labs devono redigere i seguenti documenti:

- Norme di Progetto;
- Studio di Fattibilità;
- Analisi dei Requisiti;
- Piano di Progetto;
- Piano di Qualifica.

In allegato viene consegnata anche la *Lettera di Presentazione*. Si veda la sezione 3.1.4.2 Versionamento per maggiori dettagli sul numero di versione indicato nei documenti.

2.1.1.3 Pianificazione

2.1.1.3.1 Scelta del modello di ciclo di vita Il *Responsabile di Progetto* ha il compito di scegliere il modello di ciclo di vita_G più consono allo sviluppo del prodotto richiesto, a meno che non vengano fornite indicazioni specifiche da parte del Proponente.

2.1.1.3.2 Sviluppo e documentazione del Piano di Progetto Il *Responsabile di Progetto* ha il compito di delineare i lavori che i membri del gruppo sono incaricati a eseguire. Inoltre deve calcolare costi e tempistiche per le attività da svolgere. Tali pianificazioni sono riportate nel documento *Piano di Progetto v1.0.0*.

2.2 Sviluppo

A seguire vengono esposte le attività coinvolte nella realizzazione del prodotto *software* e svolte dal gruppo Stark Labs.

2.2.1 Attività

2.2.1.1 Analisi dei requisiti Dopo aver redatto lo *Studio di Fattibilità*, gli *Analisti* hanno l'incarico di produrre il documento *Analisi dei Requisiti v1.0.0*. L'obiettivo è produrre dei requisiti a partire dalle informazioni acquisite dal gruppo. Le risorse elaborate a questo scopo provengono da uno studio del capitolato d'appalto e come risultato di eventuali incontri con Proponente e Committente.

2.2.1.2 Progettazione architetturale Per avere la definizione dell'architettura dei vari prodotti software, è necessario passare attraverso una serie di attività, quali:

- Individuare i prodotti che si intendono realizzare;
- Definire ruoli e responsabilità per ogni prodotto individuato;
- Definire le interazioni che i prodotti hanno tra di loro;
- Assicurarsi che ogni requisito sia soddisfatto da almeno uno dei prodotti individuati;
- Assicurarsi che ogni prodotto soddisfi almeno un requisito.

A questo punto, è possibile procedere con la progettazione dell'architettura dei prodotti software che devono essere realizzati. Si devono eseguire e documentare i seguenti task_G per ogni prodotto individuato:

- Suddividere il prodotto in componenti;
- Definire il ruolo di ogni componente individuato;
- Definire le interazioni tra i vari componenti;
- Definire le interfacce che ogni componente mette a disposizione dell'ambiente esterno;
- Assicurarsi che ogni requisito sia soddisfatto da almeno uno dei componenti;
- Assicurarsi che ogni componente soddisfi almeno un requisito;
- Realizzare e documentare i test di integrazione, al fine di verificare il corretto funzionamento di più componenti integrati insieme.

2.2.1.3 Progettazione di dettaglio Una volta definita l'architettura del sistema, si procede alla progettazione di dettaglio, la quale prevede che i seguenti task_G:

- Individuare le classi che implementano ciascuno dei componenti individuati mediante la progettazione architettonica;
- Definire ruoli e responsabilità per ogni classe individuata;
- Definire ogni classe nel dettaglio;
- Assicurarsi che ogni requisito sia soddisfatto da almeno una delle classi individuate;
- Assicurarsi che ogni classe soddisfi almeno un requisito.
- Realizzare e documentare i test di unità, necessari per verificare il corretto comportamento di ogni classe.

2.2.2 Norme

2.2.2.1 Classificazione dei requisiti I requisiti prodotti devono essere classificati a seconda del tipo e dell'importanza, rispettando la seguente notazione:

R[Importanza][Tipo][Codice]

- **Importanza:** i valori che può assumere sono:
 - 0: requisito obbligatorio;
 - 1: requisito desiderabile;
 - 2: requisito opzionale.
- **Tipo:** i valori che può assumere sono:
 - F: requisito funzionale;
 - P: requisito prestazionale;
 - Q: requisito di qualità;
 - V: requisito di vincolo.
- **Codice:** è il codice gerarchico e univoco del vincolo espresso nella forma X.Y.Z dove X, Y e Z sono dei valori numerici.

Inoltre, ogni requisito deve contenere le seguenti informazioni:

- **Descrizione:** descrizione del requisito con la minore ambiguità possibile;
- **Fonte:** la scelta può ricadere tra:
 - **Capitolato:** requisito ottenuto dalle specifiche del capitolato;
 - **Interno:** requisito elaborato dagli *Analisti* nel corso di un'analisi più approfondita del problema;
 - **Caso d'uso:** requisito ottenuto da uno o più casi d'uso. Deve essere quindi specificato il codice del caso d'uso a cui ci si riferisce;
 - **Verbale:** requisito ottenuto da un incontro con il Proponente o da riunioni interne tra i membri del gruppo di lavoro Stark Labs.

2.2.2.2 Classificazione dei casi d'uso I casi d'uso devono essere suddivisi in ordine gerarchico secondo il seguente schema:

UC[Codice]

- **Codice:** è il codice gerarchico e univoco che serve a identificare ogni caso d'uso.

Per ogni caso d'uso devono essere presenti anche le seguenti informazioni:

- **Titolo:** è necessario fornire un titolo riassuntivo dell'operazione che il caso d'uso intende modellare;
- **Descrizione:** è necessario fornire una breve descrizione con la minore ambiguità possibile;
- **Attori:** elenco degli attori coinvolti nel caso d'uso;
- **Scenari principali:** descrizione dei possibili scenari principali;
- **Scenari alternativi:** descrizione dei possibili scenari secondari;
- **Pre-condizioni:** una pre-condizione è una condizione sempre vera all'inizio del caso d'uso;
- **Flusso degli eventi:** ordine di esecuzione dei casi d'uso figli;
- **Inclusioni:** spiegazione di tutte le inclusioni se presenti;
- **Estensioni:** spiegazione di tutte le estensioni se presenti;
- **Generalizzazioni:** spiegazione di tutte le generalizzazioni se presenti;
- **Post-condizioni:** una post-condizione è una condizione sempre vera alla fine dell'esecuzione del caso d'uso.

Alcune fra le precedenti informazioni potrebbero essere assenti nel caso non fossero utilizzate.

2.2.2.3 Codifica dei file Tutti i *file* creati dal gruppo che contengono codice e che fanno parte della documentazione devono essere codificati tramite $UTF-8_G$ senza BOM_G . Eventuali cambiamenti di codifica devono essere approvati dal *Responsabile di Progetto*.

2.2.2.3.1 Nomi e norme di codifica Questa sezione verrà redatta nel dettaglio durante le fasi di consegna successive alla Revisione dei Requisiti. Vengono introdotte alcune norme da rispettare durante lo sviluppo del codice, indipendentemente dal linguaggio di programmazione che verrà adottato. Le norme introdotte sono le seguenti:

- In ogni *file* deve essere presente un'intestazione contenente le seguenti informazioni:
 - Versione del *file* espresso nella forma vX.Y.Z i cui dettagli sono espressi nel paragrafo 2.2.2.3.2 Versionamento;
 - Data di creazione;
 - Nome e cognome dell'autore;

- Per ogni modifica effettuata devono essere specificati: la versione successiva generata dall'avanzamento, l'autore, la data e una breve descrizione.
- I nomi delle variabili devono essere chiari, descrittivi e in inglese;
- Per le classi interne e le classi anonime deve essere aggiunta un'intestazione che ne descrive le funzionalità;
- Per ogni metodo deve essere presente un'intestazione contenente le seguenti informazioni:
 - Descrizione del metodo;
 - Lista dei parametri accompagnati da relativa descrizione;
 - Tipo di ritorno del metodo (eccetto costruttori);
 - Motivo per cui viene lanciata un'eccezione.
- Tutti i commenti vanno scritti in inglese.

2.2.2.3.2 Versionamento I file di codifica devono essere corredati dal numero di versione. Il formato adottato è il seguente:

vX.Y.Z

tale che:

- **X**: indice di versione principale. Tale valore viene incrementato ad ogni versione stabile del codice;
- **Y**: indice di modifica parziale. Tale valore viene incrementato ad ogni verifica effettuata sul codice prodotto;
- **Z**: indice di modifica minore. Tale valore viene incrementato ad ogni cambiamento che avviene sul codice.

2.2.2.3.3 Ricorsione Quando possibile, la ricorsione va evitata. Per ogni funzione ricorsiva è d'obbligo fornire una prova di terminazione. Inoltre risulta necessario valutare il costo in termini di occupazione della memoria. Nel caso in cui l'utilizzo di memoria risulti eccessivo, la ricorsione deve essere rimossa.

2.2.2.4 Inserimento dei requisiti Per tenere traccia dei requisiti è stato sviluppato un semplice servizio web accessibile ai soli membri del gruppo presso l'indirizzo

www.starklabs.altervista.org

Lo strumento realizzato si basa su un $database_G$ in MySQL_G nel quale sono memorizzate tutte le informazioni riguardanti casi d'uso e requisiti. Di seguito vengono descritte le tabelle più significative del $database$:

- **Tabella dei casi d'uso:** in essa sono presenti:
 - Nome del caso d'uso;
 - Descrizione del caso d'uso;
 - Attori relativi al caso d'uso;

- Ambito del caso d'uso, espresso in forma numerica, con 1 che rappresenta l'applicazione per gli sceneggiati, 2 l'applicazione di configurazione e 3 il modulo di sistema;
- Codice univoco che tiene traccia del caso d'uso.

- **Tabella dei requisiti:** in essa sono presenti:

- ID per tenere traccia dell'ordine di inserimento dei requisiti;
- Descrizione del requisito;
- Importanza del requisito;
- Tipologia del requisito;
- Codice del requisito, generato tramite PHP_G in modo dinamico dopo l'inserimento effettuato dall'interfaccia del servizio. Il codice può mutare nel tempo e rimane definitivo una volta che tutti i requisiti sono stati inseriti.

Nel suddetto sito è presente una pagina Requisiti dove è disponibile lo strumento creato per inserire casi d'uso e requisiti. L'interfaccia di inserimento dei casi d'uso è strutturata nel seguente modo ed è inoltre mostrata in Figura 1:

- **Attori:** area di testo per l'inserimento degli attori;
- **Nome UC:** area di testo per l'inserimento del nome del caso d'uso;
- **Codice UC:** area di testo per l'inserimento del codice del caso d'uso;
- **Visione di dettaglio dell'UC:** menu a tendina per la selezione del caso d'uso più generico da cui deriva un caso d'uso più specifico;
- **Ambito:** menu a tendina per la selezione dell'ambito del caso d'uso;
- **Descrizione:** area di testo per l'inserimento della descrizione del caso d'uso;
- **Inclusione dell'UC:** menu a tendina per la selezione del caso d'uso verso cui si fa inclusione;
- **Estende l'UC:** menu a tendina per la selezione del caso d'uso verso cui si fa estensione;
- **Commento estensione:** area di testo per l'inserimento di un commento relativo a eventuali estensioni;
- **Generalizza l'UC:** menu a tendina per la selezione del caso d'uso verso cui si fa generalizzazione.

L'interfaccia di inserimento dei requisiti è strutturata nel seguente modo ed è invece mostrata in Figura 2:

- **ID:** area di testo per l'inserimento dell'ID, con suggerimento che suggerisce l'ID subito successivo all'ultimo requisito inserito;
- **Importanza:** menu a tendina per la selezione dell'importanza;
- **Tipologia:** menu a tendina per la selezione della tipologia;
- **Descrizione:** area di testo per l'inserimento della descrizione

- **UC"x":** fino a cinque menu a tendina per selezionare le dipendenze da cui deriva il requisito, con x che va da 0 a 4;
- **Dipendenza"x":** fino a cinque menu a tendina per la selezione di dipendenze di un requisito da un altro requisito, con x che va da 0 a 4;

Inoltre, viene fornita una funzione che esporta in automatico i contenuti presenti nel *database* all'interno di un documento *LATEX*, che presenta al lettore, in modo ordinato attraverso una tabella, i requisiti inseriti nel database.

2.2.2.5 Design Pattern Per migliorare la comprensibilità delle scelte progettuali e della progettazione stessa i *Progettisti* dovranno indicare i pattern architetturali utilizzati, fornendo per ciascuno d'essi:

- Descrizione testuale;
- Descrizione grafica;
- Descrizione dell'utilizzo;
- Descrizione di come viene applicato il pattern al progetto.

2.2.2.6 Diagrammi UML Per documentare i prodotti e le componenti individuate, sarà necessario ricorrere a dei diagrammi UML_G, necessari per formalizzare gli aspetti descritti in modo testuale. Durante la progettazione architettonale, sarà necessario ricorrere a:

- **Diagrammi delle componenti:** hanno lo scopo di rappresentare la struttura interna di un prodotto software modellato mediante i suoi componenti principali e le relazioni tra di essi. Questo tipo di diagramma viene dunque utilizzato per evidenziare i componenti individuati (e le relazioni tra di loro) all'interno di ciascun prodotto software.
- **Diagrammi delle attività:** hanno lo scopo di svolgere le attività da svolgere per realizzare una certa funzionalità. Vengono dunque usati per mostrare come determinate interazioni tra componenti (o tra prodotti) realizzino una determinata funzionalità che si intende rendere disponibile.

2.2.2.7 Classificazione dei componenti I componenti vengono identificati univocamente da una descrizione nella seguente forma:

NomeProdotto::NomeComponente

dove

- **NomeProdotto** è il nome del prodotto software che i *Progettisti* hanno individuato;
- **NomeComponente** rappresenta il nome assegnato al componente dai *Progettisti*.

Inoltre ad ogni componente è associato un codice univoco nella forma

[X][Y]

dove

- **X** l'iniziale del nome del prodotto;
- **Y** è un numero incrementale.

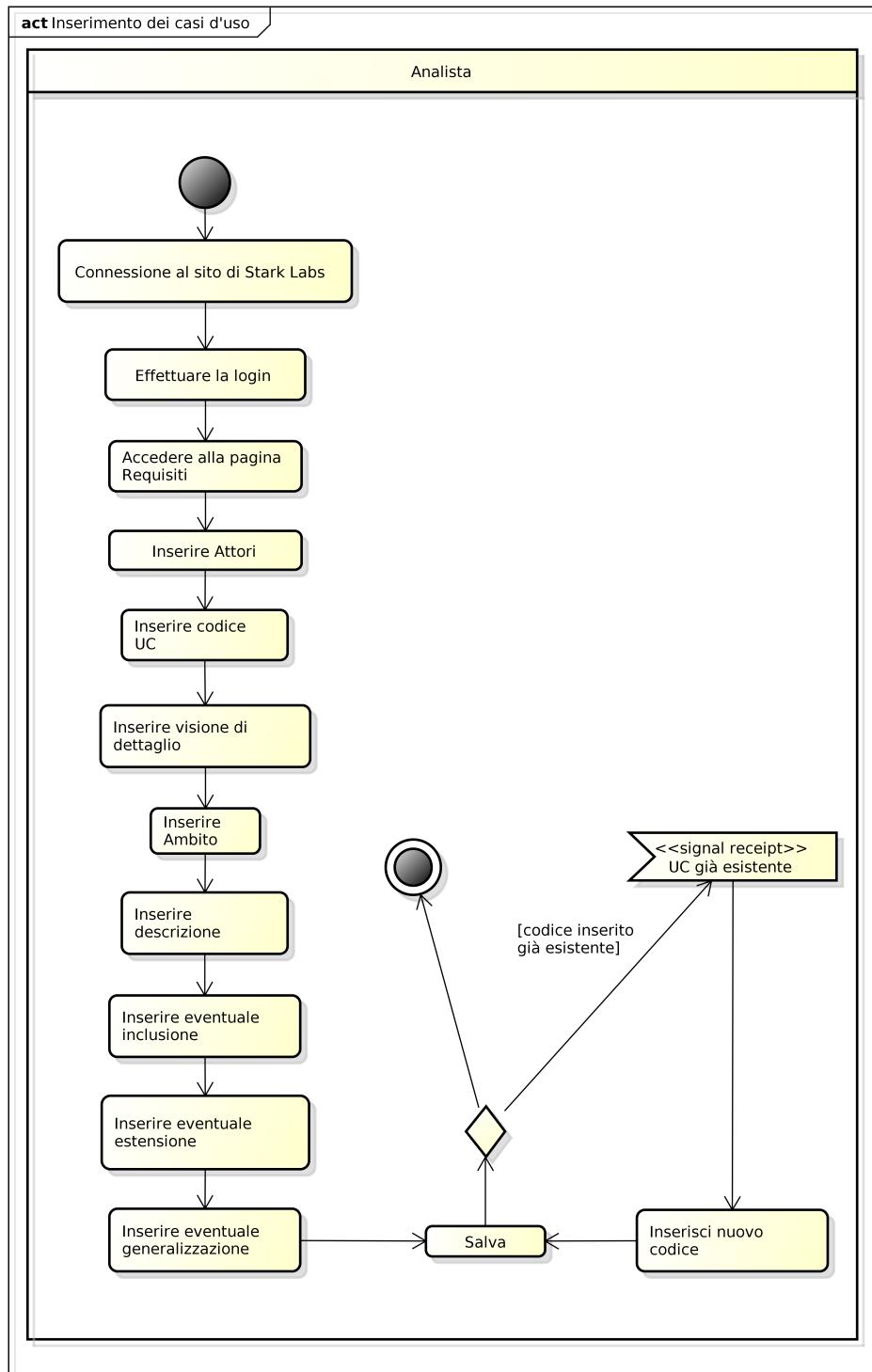


Figura 1: Diagramma di attività - Inserimento di un caso d'uso

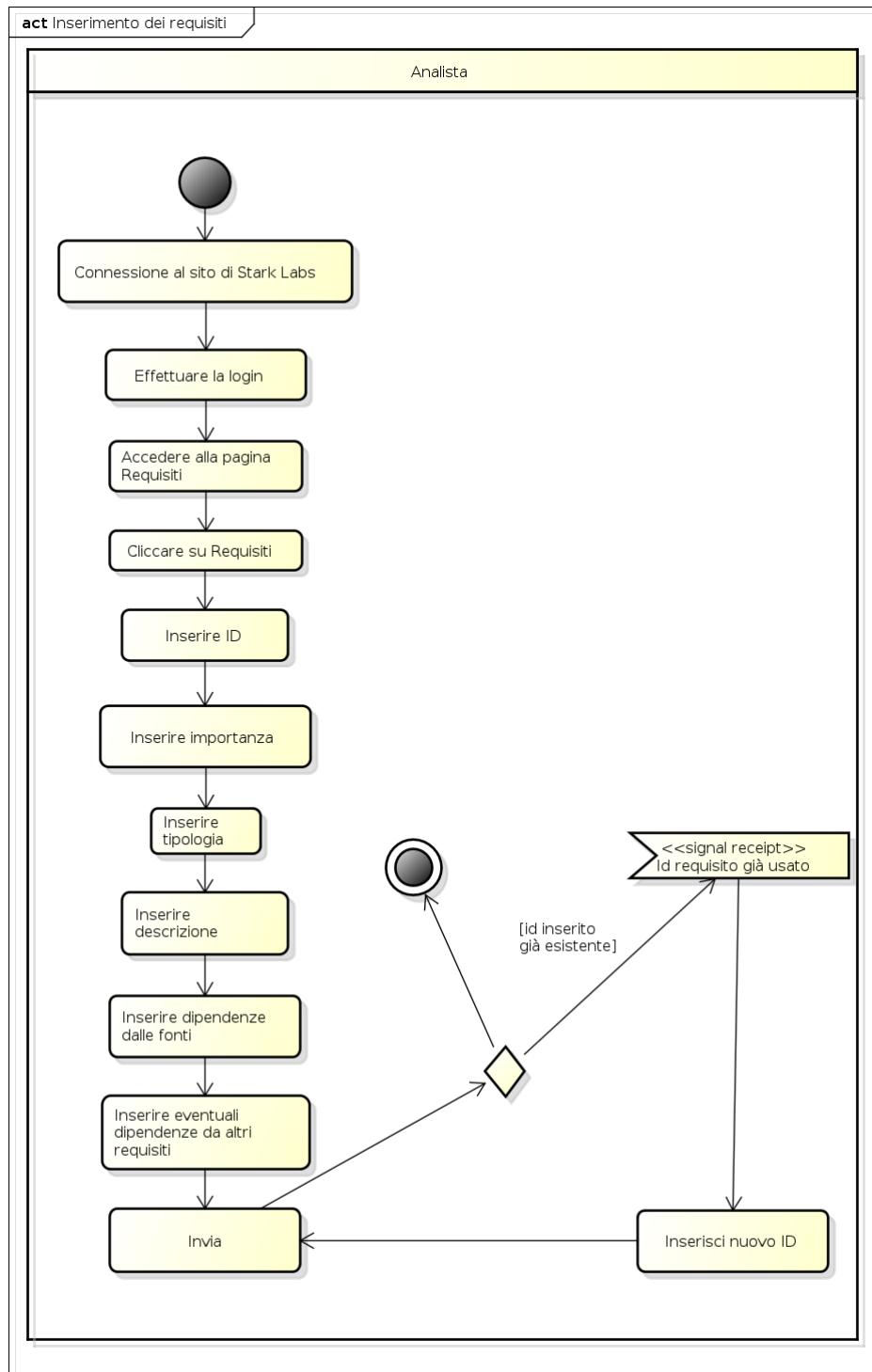


Figura 2: Diagramma di attività - Inserimento di un requisito

2.2.2.8 Classificazione delle classi Le classi vengono identificate univocamente da una descrizione nella seguente forma:

NomeProdotto::NomeComponente::NomeClasse

dove

- **NomeProdotto** è il nome del prodotto software che i **Progettisti** hanno individuato;
- **NomeComponente** rappresenta il nome assegnato al componente dai **Progettisti**.
- **NomeClasse** rappresenta il nome che è stato dato dai *Progettisti* alla classe individuata.

Inoltre ad ogni componente è associato un codice univoco nella forma

[X][Y]

dove

- X l'iniziale del nome del prodotto;
- Y è un numero incrementale.

2.2.2.9 Descrizione di una classe Per descrivere una classe, i Progettisti devono preoccuparsi di descrivere:

- nome;
- visibilità;
- attributi;
- metodi;

Inoltre, devono essere descritte dettagliatamente (anche attraverso diagrammi) le relazioni con altre classi e le interfacce messe a disposizione.

2.2.2.10 Test di unità Per la realizzazione dei test di unità i *Programmatori* devono rispettare il seguente principio: chi realizza il test non deve essere lo stesso individuo che realizza la classe da testare.

2.2.2.11 Test di integrazione Per ottimizzare l'attività di test, i **Progettisti** devono preoccuparsi di definire delle classi di verifica per accertare il funzionamento dei componenti. L'ideale sarebbe fornire degli strumenti automatici ai *Verificatori*. La progettazione delle classi per la verifica deve essere svolta rispettando il seguente principio: chi crea una classe non dovrebbe essere lo stesso individuo che crea la classe per testarla.

3 Processi di supporto

3.1 Documentazione

3.1.1 Attività

3.1.1.1 Documentazione Il gruppo di lavoro Stark Labs si impegna a registrare tutte le informazioni acquisite nel corso del ciclo di vita_G del *software* all'interno di una serie ben definita di documenti descritti in questa sezione. Nello specifico, vengono spiegati gli standard da rispettare per la stesura dei documenti e i passaggi necessari per renderli formalmente corretti.

3.1.2 Procedure

3.1.2.1 Gestione dei documenti La stesura di un nuovo documento viene decisa dal *Responsabile di Progetto*. Tutta la documentazione deve essere creata attraverso l'uso di un *template_G* \LaTeX disponibile nel *repository_G* GitHub_G del gruppo, al fine di mantenerne uniforme la struttura e lo stile.

3.1.2.2 Creazione di un nuovo documento Nel *repository_G* è presente un *file* generico denominato *new_doc.tex* contenente un *template* adattabile ad ogni nuovo documento. Il *file* deve essere incluso con il *template* \LaTeX che è messo a disposizione di tutti i membri del gruppo. Ogni redattore deve possedere tutti i *file* previsti dal *template* al fine di creare correttamente un nuovo documento.

3.1.2.3 Avanzamento di un documento La procedura adottata per sviluppare un documento è la seguente ed è riportata in Figura 3:

1. Il *Responsabile di Progetto* si preoccupa di assegnare la stesura del documento a uno o più redattori, a seconda della complessità dello stesso. L'assegnazione è gestita attraverso *task_G* organizzati tramite il servizio *online* Teamwork_G;
2. A stesura completata, ogni assegnatario ha il compito di etichettare il *task* come completato;
3. Il *Verificatore* riceve in automatico un'email che segnala il completamento del documento;
4. Nel caso si riscontrassero errori durante la fase di verifica, il *Verificatore* deve occuparsi di creare un nuovo *task* da assegnare al redattore del documento;
5. Nel caso di documento corretto, il *Verificatore* deve completare il *task* assegnatogli. Un'email viene quindi automaticamente recapitata al *Responsabile di Progetto*.
6. Il *Responsabile di Progetto* deve infine approvare il documento. In caso di mancata approvazione, il *Verificatore* deve creare un nuovo *task* indirizzato al redattore, che si occuperà di correggere gli errori in base alle segnalazioni emesse.

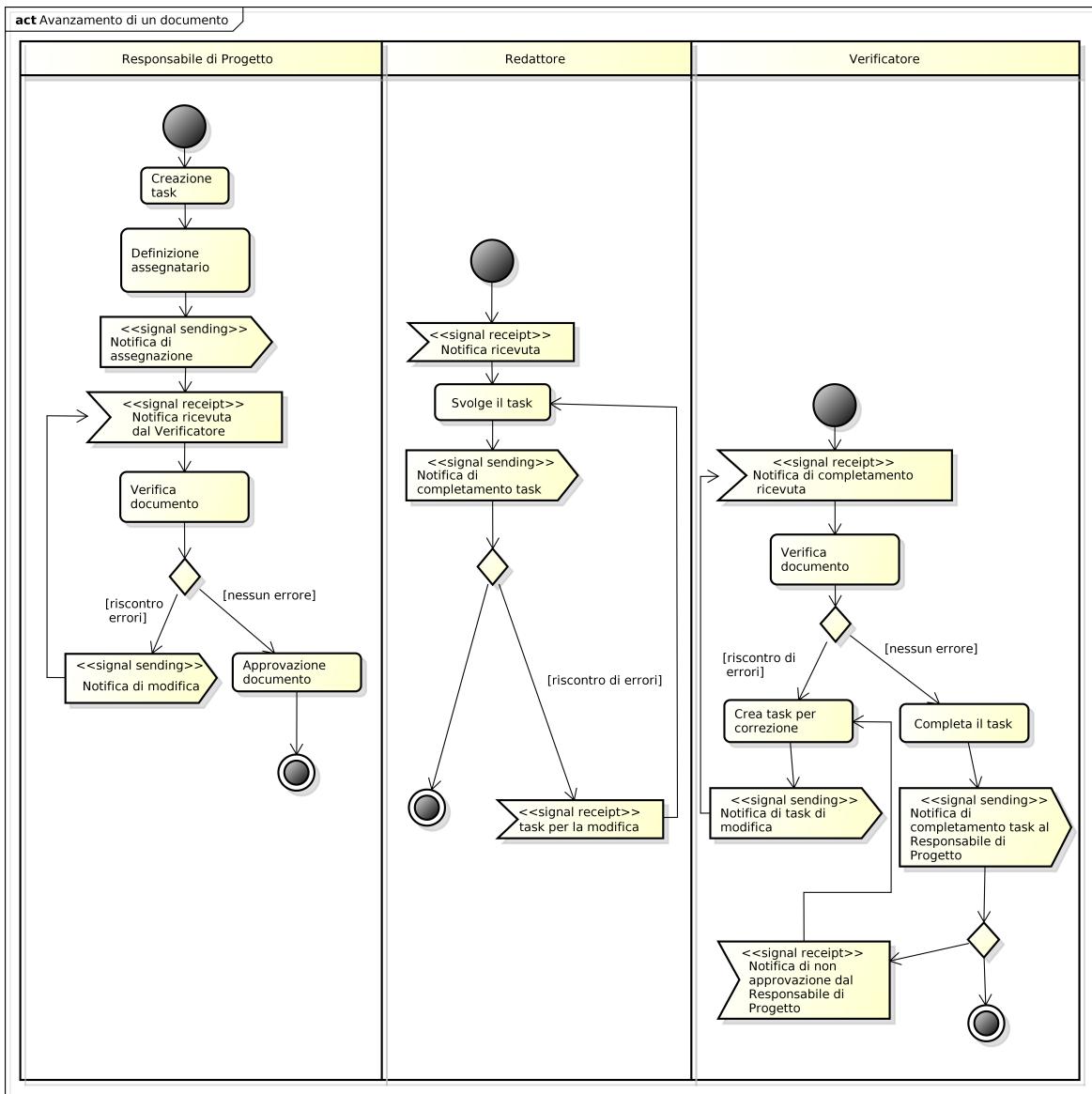


Figura 3: Diagramma di attività - Avanzamento di un documento

3.1.3 Gestione del glossario

Il popolamento del *Glossario v1.0.0* è un'attività, mostrata in Figura 4, che coinvolge redattori e *Verificatori*, e si svolge come segue:

1. **Individuazione di un nuovo termine:** se durante la stesura di un documento il redattore identifica un nuovo termine che ritiene debba essere inserito nel *Glossario v1.0.0*, è tenuto a trascriverlo all'interno di un apposito documento (*Glossario Provvisorio*) presente nella sezione *Notebooks_G* di Teamwork_G.
2. **Inserimento del termine nel Glossario:** un *Verificatore* deve occuparsi dell'approvazione di un dato termine presente in *Glossario Provvisorio* di Teamwork e dell'inserimento dello stesso all'interno del *Glossario v1.0.0*. Una volta inserito, il termine deve essere rimosso dal documento su Teamwork.

Nei documenti, il pedice _G appare solamente alla prima occorrenza del vocabolo all'interno di un paragrafo.

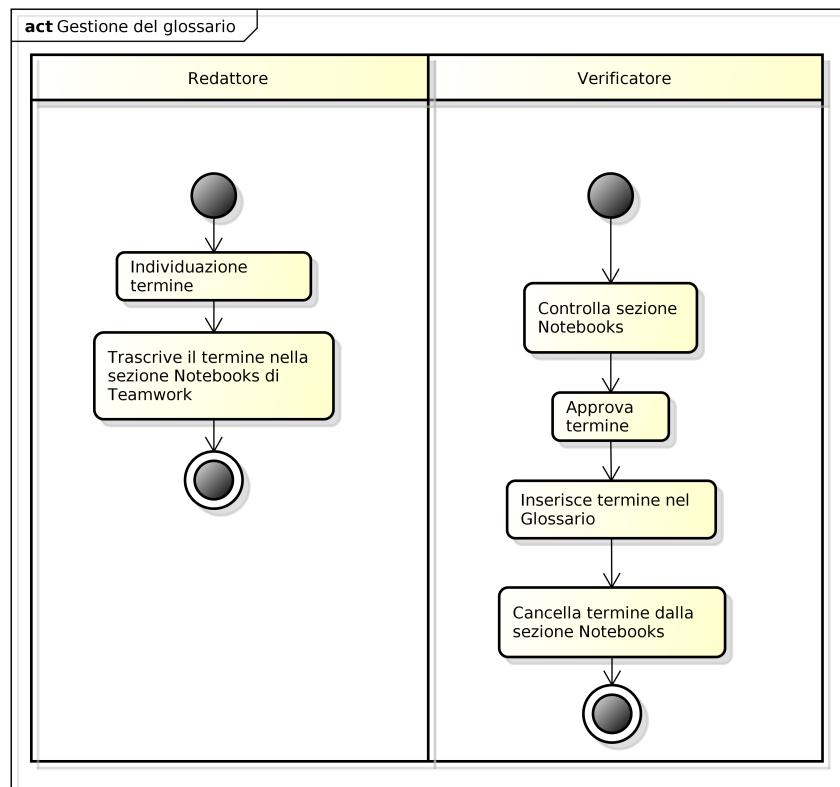


Figura 4: Diagramma di attività - Gestione del Glossario

3.1.4 Norme

3.1.4.1 Progettazione e sviluppo dei documenti Ogni documento deve rispettare in modo assoluto la seguente serie di norme.

3.1.4.2 Versionamento Ogni documento prodotto deve essere corredata dal numero di versione. Il formato adottato è il seguente:

vX.Y.Z

tale che:

- **X**: indice di versione principale. Tale valore viene incrementato ad ogni approvazione del documento e ne indica la versione di rilascio;
- **Y**: indice di modifica parziale. Tale valore viene incrementato ad ogni verifica del documento;
- **Z**: indice di modifica minore. Tale valore viene incrementato ad ogni cambiamento che avviene nel documento.

3.1.4.3 Template La creazione dei documenti avviene attraverso l'utilizzo di un *template_G* sviluppato con \LaTeX e la cui struttura **non** deve essere modificata a meno di direttive imposte dal *Responsabile di Progetto*. Il *template* funge da supporto per la stesura organizzata e sistematica dei documenti e, grazie ad esso, ogni componente del documento ha una precisa impostazione che non può essere modificata o equivocata dai redattori.

3.1.4.4 Struttura dei documenti L'organizzazione dei documenti è la seguente:

- Vi è una cartella generale in cui sono contenuti il *template_G* \LaTeX e le varie cartelle specifiche di ogni documento;
- Nella cartella denominata "template" sono contenuti i *file* di configurazione e strutturazione del *template*. Il contenuto della cartella non deve essere modificato, previa autorizzazione da parte del *Responsabile di Progetto*;
- Nella cartella specifica di ogni documento è presente un *file* di tipo nome_documento.tex, tramite cui si determina la struttura dello specifico documento. Inoltre è presente una sotto cartella "sezioni" contenente le varie sezioni nelle quali sono scritti i contenuti veri e propri.

3.1.4.4.1 Prima pagina Nella prima pagina sono presenti tutte le informazioni generali relative al documento:

- Nome del progetto;
- Logo del gruppo;
- Nome del documento;
- Versione del documento;
- Membri del gruppo che hanno lavorato come redattori, *Verificatori* e *Responsabili* per la stesura del documento. I nomi vanno scritti nel formato Nome Cognome;
- Specifica dell'uso del documento (interno o esterno);
- Lista di distribuzione del documento. I nomi vanno scritti nel formato Nome Cognome;
- Breve descrizione che precisa lo scopo del documento.

3.1.4.4.2 Registro delle modifiche Nella seconda pagina si trova una tabella con il registro delle modifiche effettuate al documento. Esso è indispensabile per un corretto tracciamento delle varie fasi che si sono percorse lungo la stesura. Ogni riga corrisponde a una modifica dove vengono segnalati:

- Descrizione dell'azione compiuta sul documento;
- Nome e cognome dell'autore della modifica;
- Data della modifica;
- Versione del documento a seguito della modifica.

3.1.4.4.3 Indici In terza pagina è presente l'indice che tiene traccia delle varie sezioni in cui è stato suddiviso il documento. La profondità dell'indice arriva fino a cinque livelli: gli argomenti trattati sono suddivisi in sezioni, sottosezioni, sotto-sottosezioni, paragrafi e sotto-paragrafi. Se sono presenti immagini e/o tabelle, viene stilato in automatico un indice che ne tiene traccia.

3.1.4.4.4 Formattazione di una pagina Tutte le pagine dei documenti seguono una precisa formattazione imposta dal $template_G$:

- **Intestazione:** contiene sulla sinistra il logo del gruppo e sulla destra il nome della sezione in cui è contenuta la pagina;
- **Contenuto:** contiene il contenuto effettivo della pagina le cui norme tipografiche sono descritte in 3.1.4.5.6 Norme Tipografiche;
- **Piè di pagina:** contiene sulla sinistra il nome del documento con relativo numero di versione e sulla destra il numero della pagina, scritto nel formato "X di Y", con X numero di pagina corrente e Y numero delle pagine totali.

La suddetta struttura si ripete per ogni pagina ad eccezione della prima, il cui formato è descritto in 3.1.4.4.1 Prima Pagina.

3.1.4.5 Suddivisione dei documenti

3.1.4.5.1 Norme di Progetto Il documento ha lo scopo di definire le linee guida per le varie attività di sviluppo. Al suo interno sono raccolte le norme, le procedure e gli strumenti che il gruppo adotterà nel corso della realizzazione del progetto. Il documento è destinato a uso interno.

3.1.4.5.2 Studio di Fattibilità Il documento ha lo scopo di descrivere le considerazioni elaborate dal gruppo per l'accettazione del progetto che si è deciso di prendere in carico, con valutazione di rischi, costi e benefici calcolati sulla base di una prima analisi del capitolo. Al suo interno vengono motivate le scelte che hanno spinto il gruppo all'esclusione degli altri progetti. Il documento è destinato a uso interno.

3.1.4.5.3 Analisi dei Requisiti Il documento ha lo scopo di identificare e descrivere i requisiti, i vincoli e gli obiettivi necessari allo sviluppo del progetto. Al suo interno sono contenuti i casi d'uso e i requisiti utili alla realizzazione del progetto, accompagnati da diagrammi e grafici di interazione fra utenti e sistema. Il documento è destinato a uso esterno.

3.1.4.5.4 Piano di Progetto Il documento ha lo scopo di pianificare lo svolgimento del progetto. Al suo interno sono fissate le risorse disponibili, la suddivisione e il calendario delle attività, e gli obiettivi necessari per valutare in modo corretto il grado di avanzamento dello sviluppo. Il documento è destinato a uso esterno.

3.1.4.5.5 Piano di Qualifica Il documento ha lo scopo di spiegare le strategie applicate al progetto per ottenere gli obiettivi di qualità. Al suo interno sono presenti le attività di verifica e pianificazione con i relativi test da sviluppare. Il documento è destinato a uso esterno.

3.1.4.5.6 Specifica Tecnica Il documento ha lo scopo di descrivere l'architettura logica del progetto, senza fissare i dettagli implementativi, ma definendo linee e strategie di realizzazione, al fine di stabilire cause ed effetti e avere una visione complessiva della soluzione. Al suo interno è contenuta una prima progettazione ad alto livello del sistema da sviluppare. In esso vengono specificati i *design pattern_G* utilizzati. Il documento è destinato a uso esterno.

3.1.4.5.7 Definizione di Prodotto Il documento ha lo scopo di descrivere nel dettaglio l'architettura del prodotto da sviluppare. Il suo contenuto viene utilizzato dai *Programmatori* per sviluppare il *software*. Il documento è destinato a uso esterno.

3.1.4.5.8 Glossario Il documento ha lo scopo di raccogliere, in ordine alfabetico, tutti i termini ambigui presenti nei documenti accompagnati da una loro definizione. Il documento è destinato a uso esterno.

3.1.4.6 Norme tipografiche Al fine di rendere omogenea e coesa la stesura dei documenti, il contenuto deve rispettare le seguenti norme tipografiche.

3.1.4.6.1 Stile del testo

- **Corsivo:** va utilizzato tassativamente per indicare termini in lingua inglese che non fanno parte dell'uso comune dell'italiano, citazioni, nomi di documenti interni, ruoli ufficiali dei membri del gruppo ed eventualmente (ma con moderazione) per parole che si ritiene debbano essere messe in risalto rispetto al resto del testo;
- **Grassetto:** va usato per evidenziare parole significative di estrema importanza. È necessario che non se ne abusi. Viene applicato automaticamente a titoli di sezioni, sottosezioni e paragrafi. Viene utilizzato negli elenchi puntati per evidenziare un termine seguito da una descrizione;
- **Maiuscolo:** viene utilizzato per scrivere acronimi e macro \LaTeX . Inoltre, lettere maiuscole vengono usate per riferirsi ai ruoli di progetto, alle fasi di lavoro e ai seguenti nomi: del *team*, del progetto e dei documenti;
- \LaTeX : viene usato il comando \LaTeX per ogni occorrenza del termine \LaTeX ;
- **Font:** nel *template* è impostato Gillius, un *font* professionale di tipo *sans-serif* il cui scopo è garantire maggiore leggibilità del testo su schermo;
- **Monospace_G:** tipologia di font che serve per formattare correttamente il testo contenente porzioni di codice, comandi e indirizzi web.

3.1.4.6.2 Punteggiatura

- **Spaziatura:** lo spazio non può mai precedere un carattere di punteggiatura.

3.1.4.6.3 Composizione del testo

- **Elenchi puntati:** l'ultima voce deve terminare con un punto, mentre le altre con un punto e virgola. La prima lettera di ogni punto va scritta in maiuscolo e la prima parola va in grassetto se seguita da una descrizione della stessa. Gli elenchi numerati vengono utilizzati per descrivere sequenze di azioni ordinate;
- **Pedice G:** il pedice G è utilizzato al solo scopo di indicare termini potenzialmente ambigui che si possono reperire nel documento *Glossario v1.0.0*.

3.1.4.6.4 Formati Elenco dei formati rispettati dai documenti:

- **Data:** il formato utilizzato è dd/mm/yyyy. Per esempio, 16/11/2004 indica il 16 novembre 2004;
- **Ora:** si utilizza il formato internazionale previsto dalla norma ISO 8601 del tipo [hh]:[mm]:[ss] ove [hh] indica l'ora, [mm] i minuti, [ss] i secondi, espressi con due cifre. Ad esempio, 18:35:26 indica le ore 18, 35 minuti e 26 secondi; 04:09:01 indica le ore 4, 9 minuti e 1 secondo;
- **Percorsi:** viene utilizzato il separatore *slash* (/) per indicare il percorso di un *file*. Per esempio, cartella1/cartella2/file_esempio.txt;
- **Nomi di persona:** espressi nel formato Nome Cognome.

3.1.4.6.5 Sigle Elenco delle sigle che possono apparire nel corso dei documenti:

- **AdR:** Analisi dei Requisiti;
- **GL:** Glossario;
- **NdP:** Norme di Progetto;
- **PdP:** Piano di Progetto;
- **PdQ:** Piano di Qualifica;
- **SdF:** Studio di Fattibilità;
- **ST:** Specifica Tecnica;
- **RA:** Revisione di Accettazione;
- **RP:** Revisione di Progettazione;
- **RQ:** Revisione di Qualifica;
- **RR:** Revisione dei Requisiti.

3.1.4.7 Componenti grafiche

3.1.4.7.1 Immagini e diagrammi Tutte le immagini (diagrammi inclusi) da inserire nei documenti devono essere salvate in formato *Portable Network Graphics* (PNG_G) o *Portable Document Format* (PDF_G). Immagini e diagrammi vanno inserite nella cartella "immagini" relativa al documento specifico.

3.2 Qualità

3.2.1 Test

3.2.1.1 Classificazione dei test I Programmatori si occupano di creare test di unità e integrazione, mentre gli Analisti creano quelli di sistema e di validazione, al fine di verificare che tutti i requisiti individuati dagli Analisti nel documento *Analisi dei Requisiti v2.0.0* siano soddisfatti. Ogni test deve essere codificato come:

T[X][Y]

dove:

- X indica il tipo di test. I valori possibili sono:
 - V: indica un test di validazione;
 - S: indica un test di sistema;
 - I: indica un test di integrazione;
 - U: indica un test di unità.
- Y indica il codice univoco di ogni test che corrisponde al livello gerarchico del requisito corrispondente per i test di validazione e sistema, mentre per i test di integrazione e unità è un valore numerico crescente (a partire da 1).

3.2.2 Metriche

3.2.2.1 Classificazione delle metriche Sono state definite delle metriche, riportate nel *Piano di Qualifica v2.0.0* al fine di garantire la qualità dei processi:

M[X][Y][Z]

dove:

- X indica se la metrica si riferisce a processi o prodotti e può assumere i valori:
 - PC per indicare i processi;
 - PD per indicare i prodotti.
- Y presente solo se la metrica è riferita ai prodotti. Specifica se il prodotto è un documento o parte del *software* e può assumere i seguenti valori:
 - D per indicare i documenti;
 - S per indicare il *software*.
- Z indica il codice univoco della metrica (un numero incrementale a partire da 1).

In sintesi, le tre possibilità di denominazione sono le seguenti:

- MPC[Z];
- MPDD[Z];
- MPDS[Z].

3.2.2.2 Procedure

3.2.2.2.1 Calcolo delle metriche attraverso Android Studio Per calcolare i valori di alcune metriche per il *software* è necessario l'utilizzo di *Android Studio_G* e di un suo plugin, chiamato *MetricsReloaded_G*. Di seguito viene mostrata la procedura da seguire in queste situazioni:

1. Aprire *Android Studio_G*;
2. Dal menù in alto selezionare *Analyze* e successivamente *Calculate Metrics*;
3. Selezionare in *Metrics scope* il box *custom scope*;
4. Nel menù a tendina a fianco selezionare i file su cui effettuare il calcolo delle metriche;
5. Selezionare in *Metrics profile* il pulsante a fianco del menu identificato da '...';
6. Nella finestra ora aperta selezionare tramite checkbox la metrica da calcolare:
 - **Nested block depth**: per rappresentare il numero di livelli di annidamento dei metodi;
 - **Class size (attributes)**: per rappresentare il numero di attributi per una classe;
 - **Number of parameters**: per rappresentare il numero di parametri per metodo;
 - **Comment Ratio**: per rappresentare il numero di linee di codice per linee di commento.
7. Selezionare *Apply* e in seguito *OK*;
8. Selezionare nuovamente *OK*;

3.2.2.3 Strumenti

3.2.2.3.1 Copertura dei requisiti obbligatori Lo strumento scelto per il calcolo del valore di questa metrica è *Stark Labs Software*, che permette di tracciare i requisiti e mapparli su use case, componenti e classi.

3.2.2.3.2 Copertura dei requisiti desiderabili Lo strumento scelto per il calcolo del valore di questa metrica è *Stark Labs Software*, che permette di tracciare i requisiti e mapparli su use case, componenti e classi.

3.2.2.3.3 Numero livelli di annidamento Lo strumento scelto per il calcolo del valore di questa metrica è *Metrics Reloaded_G*, plugin di *Android Studio_G*.

3.2.2.3.4 Numero di attributi per classe Lo strumento scelto per il calcolo del valore di questa metrica è *Metrics Reloaded_G*, plugin di *Android Studio_G*.

3.2.2.3.5 Numero di parametri per metodo Lo strumento scelto per il calcolo del valore di questa metrica è *Metrics Reloaded_G*, plugin di *Android Studio_G*.

3.2.2.3.6 Linee di codice per commento Lo strumento scelto per il calcolo del valore di questa metrica è *Metrics Reloaded_G*, plugin di *Android Studio_G*.

3.2.2.3.7 Copertura di codice Lo strumento scelto per il calcolo del valore di questa metrica è *Android Studio_G*.

3.2.2.3.8 Percentuale di test superati Lo strumento scelto per il calcolo del valore di questa metrica è *Stark Labs Software*.

3.2.2.3.9 Failure avoidance Lo strumento scelto per il calcolo del valore di questa metrica è il *Framework_G Espresso_G*, fornito dalla *Android_G Testing Support Library*.

3.2.2.3.10 Breackdown avoidance Lo strumento scelto per il calcolo del valore di questa metrica è *Framework_G Espresso_G*, fornito dalla *Android_G Testing Support Library*.

3.2.3 Obiettivi

3.2.3.1 Classificazione degli obiettivi Sono state definiti degli obiettivi, riportati nel *Piano di Qualifica v2.0.0* al fine di garantire la qualità dei processi:

OQ[X][Y][Z]

dove:

- X indica se la metrica si riferisce a processi o prodotti e può assumere i valori:
 - PC per indicare i processi;
 - PD per indicare i prodotti.
- Y presente solo se la metrica è riferita ai prodotti. Specifica se il prodotto è un documento o parte del *software* e può assumere i seguenti valori:
 - D per indicare i documenti;
 - S per indicare il *software*.
- Z indica il codice univoco della metrica (un numero incrementale a partire da 1).

In sintesi, le tre possibilità di denominazione sono le seguenti:

- OQPC[Z];
- OQPDD[Z];
- OQPDS[Z].

3.3 Verifica

La verifica consiste nel controllare che il materiale prodotto al raggiungimento delle *milestone_G* sia conforme agli obiettivi prefissati. Pertanto, è necessario effettuare delle verifiche per controllare che non siano stati commessi errori. Una corretta applicazione del processo di verifica genera un aumento del rapporto fra efficienza ed efficacia, riducendo il tempo impiegato nel percorso di analisi e correzione.

3.3.1 Attività

3.3.1.1 Analisi statica L'attività di analisi statica è una tecnica di verifica applicabile sia a documenti che a codice sorgente che va ad analizzare il solo testo del *file* senza mandarlo in esecuzione. Tale tecnica viene utilizzata durante l'intero sviluppo del progetto e si pone come obiettivo il ritrovamento di eventuali anomalie. I metodi di controllo sono i seguenti:

- **Walkthrough:** si ricercano all'interno di testo o codice tutte le possibili anomalie; l'attività è eseguita da persone. L'analisi si basa sulla lettura di tutto il contenuto del *file*. In seguito al ritrovamento di anomalie, le si analizza con il redattore del documento (o *Programmatore* nel caso di codice) e si indaga per raggiungere una soluzione del problema. Tale tecnica risulta utile durante le prime fasi di sviluppo del progetto, in quanto manca, ai componenti, una visione complessiva del documento o del codice che si sta scrivendo. Permette così ai *Verificatori*, dopo aver svolto le prime correzioni, di preparare una *lista di controllo* con gli errori più frequenti in modo da migliorare l'efficienza delle verifiche future. A questo punto è possibile implementare un'analisi mirata e più efficiente attraverso il metodo dell'*Inspection*.
- **Inspection:** si ricercano all'interno di un testo errori specifici; l'attività può essere eseguita sia da un umano che, nel caso di anomalie nella sintassi, da uno script. Il metodo focalizza la ricerca su errori presupposti identificati dalla *lista di controllo*.

3.3.1.2 Analisi dinamica L'attività di analisi dinamica è una tecnica di verifica applicabile solamente al *software*. Tale tecnica può essere utilizzata per analizzare l'intero *software* o una porzione limitata dello stesso. L'attività consiste nell'esecuzione di *test* automatici realizzati dal *team*. Le verifiche devono essere effettuate su un insieme finito di casi, con valori di ingresso, uno stato iniziale e un esito decidibile. Tutti i *test* producono risultati automatici che inviano notifiche sulla tipologia di problema individuato. Ogni *test* è ripetibile, ossia applicabile durante l'intero ciclo di vita_G del *software*. Le caratteristiche da rispettare sono le seguenti:

- **Ambiente:** è necessario riportare l'ambiente sia *software* che *hardware* in cui il sistema esegue il *test*. Deve essere specificato lo stato iniziale del sistema;
- **Specifiche:** è necessario riportare i dati in ingresso e in uscita per verificare l'esito del *test*, ossia se il codice analizzato è conforme alle aspettative;
- **Procedure:** è possibile specificare ulteriori istruzioni per l'esecuzione dei *test*. Inoltre possono essere riportate istruzioni sulla corretta lettura dei risultati.

3.3.1.3 Gestione anomalie Se si dovessero riscontrare anomalie o discordanze normative durante le attività di verifica, il *Verificatore* ha il dovere di notificarle all'assegnatario del *task*_G.

3.3.1.4 Tracciamento L'attività di tracciamento svolta dai *Verificatori* consiste nella catalogazione di tutti i casi d'uso e dei requisiti che ne derivano ed evidenziare la corrispondenza fra di essi in modo da avere ben chiaro i processi di derivazione.

3.3.2 Procedure

3.3.2.1 Procedure di assegnazione delle anomalie L'assegnazione delle anomalie rilevate dai *Verificatori* avviene attraverso l'uso del servizio *web Teamwork*_G. Gli errori vanno gestiti attraverso *task*_G da creare in una delle seguenti categorie:

- **Anomalie Documenti:** insieme di *task* dedicato agli errori rilevati nella documentazione;
- **Anomalie Codice:** insieme di *task* dedicato agli errori rilevati nel codice.

I *task* contenuti nei suddetti gruppi devono essere creati seguendo la seguente serie di azioni, mostrate anche in Figura 5:

1. Assegnazione di un titolo al nuovo *task* contenente in testa la dicitura [BUG];
2. Descrizione dettagliata dell'anomalia rilevata. Deve essere specificato il punto esatto del file in cui è stato individuato l'errore;
3. Assegnazione del *task* al membro del gruppo che ha commesso l'errore;
4. Assegnazione di una data di scadenza per la correzione dell'errore.

Una volta creato il *task*, un'email viene inviata automatica all'assegnatario, che avrà l'incarico di risolvere il bug_G entro la data di scadenza segnata. La priorità di risoluzione degli errori deve essere svolta secondo l'ordine di scadenza organizzato dal *Responsabile di Progetto*.

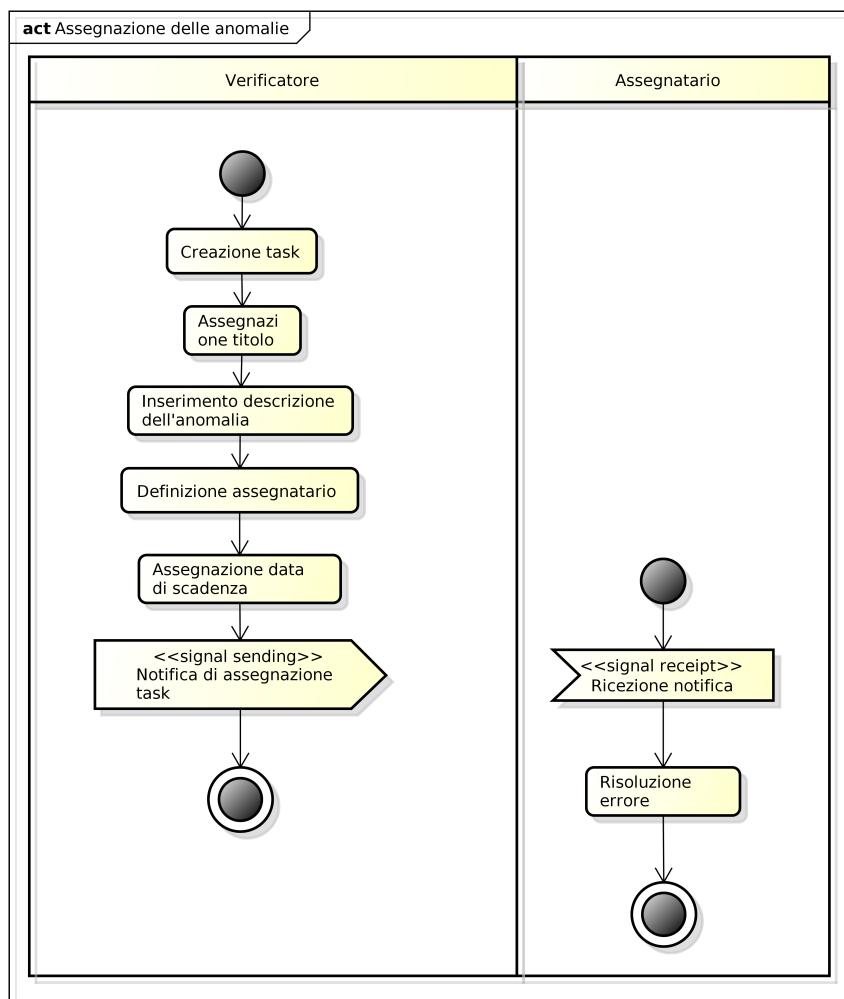


Figura 5: Diagramma di attività - Procedura di assegnazione delle anomalie

3.3.3 Strumenti

3.3.3.1 Correzione ortografica Come supporto alla correzione dei documenti si utilizza il correttore ortografico automatico incluso nell'*editor TeXstudio_G*. Gli strumenti offerti da TeXstudio sono utili alla sola correzione di errori ortografici gravi, ma non è preciso per l'individuazione di errori più sottili. Un'analisi più approfondita spetta ai *Verificatori*.

3.3.3.2 Calcolo dell'indice Gulpease Affinchè un documento possa superare la fase di accettazione, è necessario che soddisfi il test di leggibilità con un indice Gulpease_G superiore a 40 punti. Per maggiori dettagli si veda il documento *Piano di Qualifica v1.0.0*.

3.4 Validazione

Il prodotto è validato se il risultato ottenuto è consistente e conforme verso le attese del Proponente e la pianificazione svolta dal team.

3.4.1 Responsabilità

I ruoli delegati alla validazione del prodotto sono i seguenti:

- *Verificatori*: hanno il compito di eseguire i test e tracciarne i risultati per effettuarne una valutazione in relazione alle metriche stilate nel *Piano di Qualifica*.
- *Responsabile di Progetto*: valuta i risultati ottenuti dai test e decide se approvarli o meno. Ha quindi il compito di riportare al Proponente i risultati ottenuti fornendo indicazioni sull'esecuzione dei test di accettazione per consentire la validazione del prodotto finale.

3.5 Procedure

3.5.1 Procedura per la validazione

Vengono riportati di seguito i passi da seguire per effettuare la validazione sul prodotto:

1. I *Verificatori* eseguono i test e ne traggono i risultati per effettuare una valutazione in relazione alle metriche stilate nel *Piano di Qualifica*.
2. Il *Responsabile di Progetto* valuta i risultati ottenuti e decide se:
 - Accettare i risultati;
 - Richiedere una nuova esecuzione dei test con l'obiettivo di ottenere risultati migliori.
3. A seguito dell'accettazione dei test, il *Responsabile di Progetto* ha il compito di consegnarne i risultati al Proponente, fornendo indicazioni sull'esecuzione dei test di accettazione per consentire la validazione del prodotto finale.

4 Processi organizzativi

4.1 Gestione dei processi

4.1.1 Attività

4.1.1.1 Gestione delle comunicazioni

4.1.1.1.1 Comunicazione interna Viene utilizzato un gruppo Telegram_G per comunicare informalmente fra i membri del *team*. Il servizio fornisce il vantaggio di essere un'applicazione multipiattaforma_G e disponibile nelle versioni *desktop*, *web* e *mobile*. Inoltre è possibile catalogare e tenere traccia degli argomenti tramite l'uso di *hashtag*_G (per esempio: #analisiRequisiti), e chiamare all'attenzione un dato membro attraverso l'utilizzo del comando chiocciola (per esempio: @GordonFreeman).

4.1.1.1.2 Comunicazione esterna Il *Responsabile di Progetto* è la persona preposta a mantenere i contatti con individui esterni al gruppo. Per tali comunicazioni è stato creato il seguente indirizzo di posta elettronica:

starklabs.swe@gmail.com

Tutti i componenti del gruppo possono accedere all'indirizzo di posta, tuttavia solo il *Responsabile di Progetto* può inviare comunicazioni con tale indirizzo email. Tutte le email ricevute verranno automaticamente inoltrate agli indirizzi personali dei membri del *team*.

4.1.1.1.3 Composizione email

- **Destinatario:**

- I destinatari possono essere il Proponente (Giulio Paci e l'azienda MIVOQ s.r.l.), il Prof. Tullio Vardanega e il Prof. Riccardo Cardin.

- **Mittente:**

- L'unico indirizzo utilizzabile è starklabs.swe@gmail.com e può essere usato solamente dal *Responsabile di Progetto*.

- **Oggetto:** l'oggetto deve contenere la dicitura [UNIPD-TTS] così come è stato specificato nel capitolo dell'azienda Proponente. Nel caso il messaggio sia una risposta è consigliabile aggiungere la particella “Re:” all'inizio dell'oggetto per rendere chiara la distinzione del livello di risposta; se si dovesse trattare di un inoltro si deve usare la particella “l:”. L'oggetto non va mai cambiato.

- **Corpo:** in caso di risposta da parte dell'azienda MIVOQ o del Committente, risulta utile la citazione della frase a cui si intende rispondere. Il modello per citare correttamente una porzione del messaggio deve rispettare le seguenti regole: devono essere presenti data e ora della mail a cui si risponde, il nome del mittente e il suo indirizzo email tra parentesi angolari. Per esempio: <starklabs.swe@gmail.com>, la dicitura “ha scritto:” e infine il testo con all'inizio una parentesi angolare chiusa (“>testo di prova”). Se dovessero essere presenti alcune parti con uno o più destinatari specifici, il nome dovrà essere indicato all'inizio del paragrafo attraverso la dicitura: @destinatario.

- **Allegati:** qualora vi fosse necessità, è possibile allegare alcuni *file* al messaggio email. Possono per esempio essere allegati i verbali di eventuali incontri con Proponente o Committente, oppure *file* facenti parte della documentazione spiegata in sezione 3.1.4.5 Suddivisione dei documenti.

4.1.1.2 Gestione delle riunioni

4.1.1.2.1 Riunioni interne

- **Frequenza:** le riunioni del gruppo di lavoro avranno cadenza settimanale;
- **Convocazione:** Il *Responsabile di Progetto* ha il compito di convocare le riunioni generali, a cui dovranno partecipare tutti i membri del gruppo. Su decisione del *Responsabile di Progetto* le riunioni possono coinvolgere anche solo specifici componenti del gruppo, a seconda del ruolo che si ritiene più utile in una data fase del progetto. Al termine di ogni riunione viene redatto un verbale. Il *Responsabile* deve convocare l'assemblea con almeno un giorno di preavviso attraverso l'invio di una comunicazione ufficiale nel gruppo *Telegram_G*, messa in rilievo tramite l'uso dell'*hashtag_G #riunioneDATA*, con DATA espressa nel formato d/mmmm/yyyy. Per esempio: *#riunione8marzo2016* raccolgono i messaggi inerenti alla riunione tenutasi in data 8 marzo 2016. Nel corpo del messaggio deve essere specificato:
 - **Data:** data e ora prevista;
 - **Luogo:** luogo previsto;
 - **Tipo:** ordinaria o straordinaria;
 - **Ordine del giorno:** elenco ordinato delle voci da esaminare.

Ogni componente del gruppo deve rispondere al messaggio nel più breve tempo possibile, confermando la presenza o giustificando l'eventuale assenza. In assenza di una risposta di uno o più membri entro 24 ore, il *Responsabile di Progetto* ha il compito di contattare telefonicamente gli interessati. Una volta ricevute le risposte e verificata l'assenza o presenza dei membri convocati, il *Responsabile di Progetto* ha la possibilità di decidere se confermare o posticipare la riunione, in modo da permettere la presenza di tutti i membri chiamati; tutte le eventuali modifiche dovranno essere notificate tramite lo stesso *hashtag* utilizzato per organizzare la riunione.

- **Verbale:** il verbale di riunione interna si presenta in forma di documento informale, utile al solo scopo di fissare in modo ordinato i punti principali trattati e le relative soluzioni proposte. Il verbale deve essere redatto come documento testuale utilizzando la funzione *Notebooks_G* di *Teamwork_G*. In questo modo si facilita la condivisione, tra tutti i membri del gruppo, di un documento mantenuto aggiornato dal segretario della riunione. Tale ruolo viene scelto a rotazione tra i membri convocati. È inoltre compito del segretario annotare ogni argomento trattato e controllare che venga rispettato l'ordine del giorno.

4.1.1.2.2 Riunioni esterne

- **Convocazione:** in questo caso viene utilizzata l'email *starklabs.swe@gmail.com* attraverso la quale il *Responsabile di Progetto* si occupa di contattare l'azienda Proponente e di mettere in cc_G i membri del gruppo. Per quanto sia auspicabile una riunione

plenaria, eventuali assenze dei componenti del *team* non causeranno posticipazioni o spostamenti delle date di incontro, dovendo ovviamente considerare gli impegni dell'azienda stessa.

- **Verbale:** in caso di riunione con il Committente o il Proponente, il verbale è un documento che assume carattere ufficiale e che pertanto viene redatto secondo uno schema specifico. Per agevolare la scrittura di tale documento viene utilizzato un *template_G* *LATEX* per definire la struttura e organizzare i contenuti. Tale documento dovrà essere redatto e inviato come allegato in risposta all'email di convocazione dell'assemblea al Proponente Giulio Paci direttamente dal segretario scelto tra i membri presenti.

4.1.1.3 Gestione del sistema dei task Il sistema selezionato per la gestione dei *task_G* è Teamwork_G, un servizio web di *project management_G*. Le viste presenti sono:

- **Dashboard:** dove vengono visualizzati i progetti attivi e le ultime notizie relative ad essi;
- **Everything:** che consente di visualizzare i *task*, le *milestone_G* e i *file* con possibilità di filtrarli per data;
- **Project:** permette di visualizzare la lista di tutti i progetti suddivisi per categoria e ne consente l'accesso;
- **Calendar:** mostra un calendario per la gestione degli impegni e delle scadenze;
- **Statuses:** consente di verificare gli stati dei collaboratori del progetto;
- **People:** permette di visualizzare l'elenco dei singoli elementi del gruppo di lavoro e di accedere al loro profilo.

Le funzionalità principali si hanno in seguito all'accesso al progetto desiderato. Esse si suddividono nei seguenti punti:

- Aggiunta di nuovi *task*, ed eventualmente di sotto *task*, da associare ad uno o più membri del *team*;
- Assegnazione a ciascun *task* di una data d'inizio e di completamento;
- Aggiunta di nuove *milestone* con relativi dettagli come responsabile, descrizione e data di scadenza;
- *Upload_G* di file potenzialmente utili al gruppo di lavoro;
- Utilizzo di un blocco note.

4.1.1.4 Gestione delle milestone Il *Responsabile di Progetto* ha il compito di pianificare i punti di controllo che il *team* deve raggiungere, assicurandosi che ogni *task_G* necessario al suo soddisfacimento venga terminato entro la data prestabilita.

4.1.1.5 Gestione dei task È compito del *Responsabile di Progetto* individuare ogni singolo *task_G* e, al seguito di un'accurata valutazione, assegnarlo al membro del gruppo più adatto. Deve inoltre associare una data di inizio e di scadenza di fattibilità. Tutte queste attività possono essere facilmente accedute attraverso l'interfaccia grafica di Teamwork_G.

4.1.1.6 Gestione dello svolgimento dei task Ogni membro del gruppo di lavoro è tenuto ad accettare il $task_G$ assegnatogli dal *Responsabile di Progetto* e fare quanto possibile per portarlo a termine entro la data di scadenza. Nel caso in cui l'assegnatario non fosse in grado di adempire al suo compito è tenuto a renderlo noto al *Responsabile di Progetto* entro 24 ore dall'assegnazione del *task*, altrimenti quest'ultimo verrà considerato come accettato; solo dopo un'accurata valutazione delle motivazioni riportate, il *Responsabile di Progetto* può provvedere a trovare un nuovo membro da incaricare allo svolgimento del *task*.

4.1.2 Procedure

4.1.2.1 Generazione di una milestone Dopo aver avuto accesso al progetto di Teamwork_G, il *Responsabile di Progetto* deve eseguire i seguenti passi per generare una $milestone_G$:

1. Cliccare sul pulsante "Add a milestone";
2. Definire il titolo della *milestone*;
3. Definire la data di scadenza;
4. Assegnarla ad un responsabile.

4.1.2.2 Assegnazione di un task Dopo essere acceduto al progetto di Teamwork_G, il *Responsabile di Progetto* deve eseguire i seguenti passi, mostrati anche in Figura 6, per generare un $task_G$:

1. Cliccare sul pulsante "Add a task";
2. Definire il titolo del *task*;
3. Definire la data di inizio;
4. Definire la data di scadenza;
5. Assegnarlo ad uno o più membri del gruppo.

4.1.2.3 Svolgimento di un task Il membro assegnatario del $task_G$, ricevuta la notifica e non avendo alcun impedimento, deve procedere secondo le seguenti direttive, mostrate anche in Figura 7:

1. Se il *task* ricevuto ha una scadenza più immediata rispetto a quello su cui sta lavorando, deve sospendere lo svolgimento di quest'ultimo, metterlo in coda e dedicarsi al *task* appena notificato;
2. Se, dopo aver iniziato lo svolgimento del *task*, si riceve la notifica di uno nuovo con scadenza più immediata si procede come riportato nel punto precedente;
3. Se si dovesse superare la data di scadenza prevista, è necessario impostare il *tag_G* "Delay" dal sistema offerto su Teamwork_G. Questa situazione si può verificare se:
 - Il tempo assegnato dal *Responsabile di progetto* non è sufficiente al completamento del *task*;
 - Il *task* in ritardo sta alle dipendenze di un altro *task* non ancora completato;

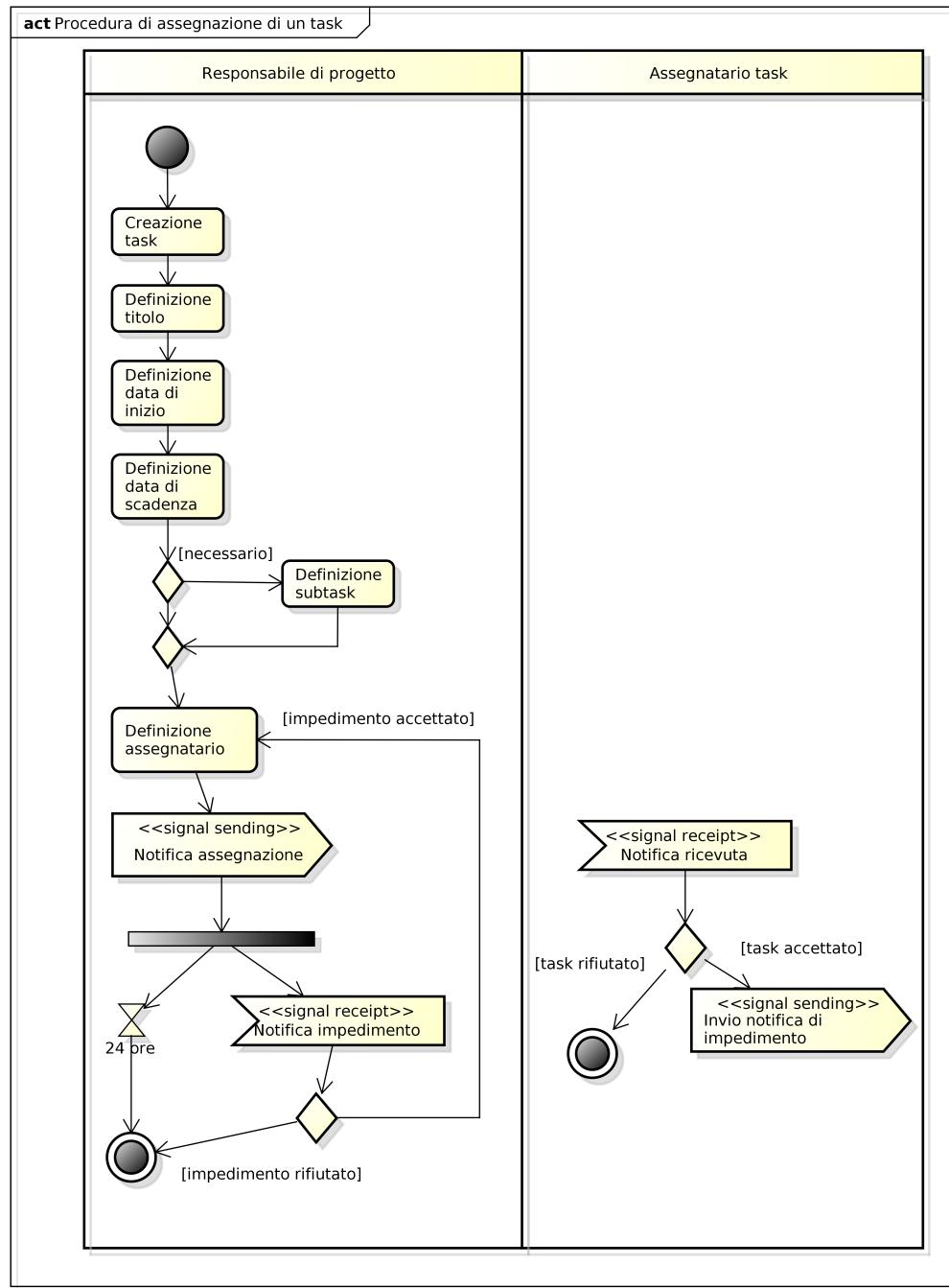


Figura 6: Diagramma di attività - Procedura di assegnazione di un task

- L'assegnatario è rallentato da cause esterne non rese note al *Responsabile di Progetto*;
- Il membro incaricato non ha a disposizione tutte le conoscenze necessarie per un corretto svolgimento del *task*.

Spetta al *Responsabile di Progetto* fare in modo che i primi due casi non si verifichino.

4. Al completamento del lavoro l'assegnatario deve spuntare il *task* dalla lista presente su Teamwork_G;
5. A questo punto può proseguire con lo svolgimento dei *task* rimanenti riprendendo la procedura dall'inizio.

4.1.2.4 Rilevamento dei rischi È compito del *Responsabile di Progetto* individuare i rischi trovati nel *Piano di Progetto v1.0.0*. Questa attività necessita di un continuo monitoraggio, in quanto è plausibile che insorgano nuovi rischi in seguito a quelli rilevati nella fase preliminare. In tal caso il *Responsabile di Progetto* deve agire come segue:

1. Registrare il resoconto effettivo dei rischi nel *Piano di Progetto v1.0.0*;
2. Pianificare per gestire i nuovi rischi;
3. Aggiornare le metodologie per far fronte alla nuova pianificazione;
4. Monitorare i nuovi rischi riscontrati durante lo sviluppo del progetto.

4.1.3 Norme

4.1.3.1 Ruoli di Progetto Ogni componente del gruppo Stark Labs deve ricoprire almeno una volta ciascuno dei ruoli necessari allo sviluppo del progetto. Nell'assegnazione dei compiti il *team* si impegna a distribuire equamente le ore di lavoro previste per ogni ruolo. Questo criterio ha guidato la stesura del *Piano di Progetto v1.0.0* di cui si rimanda alla lettura. Di seguito vengono presentati i diversi incarichi, delineando per ciascuno mansioni e responsabilità.

4.1.3.1.1 Responsabile di Progetto Il *Responsabile di Progetto* rappresenta il *team* e il progetto nei confronti di Committente e Proponente; accentra le responsabilità di scelta e approvazione. Detiene inoltre le seguenti responsabilità:

- Pianificazione e coordinamento delle attività;
- Gestione e controllo delle risorse;
- Analisi e gestione dei rischi;
- Approvazione dei documenti;
- Assicurarsi che tutte le attività svolte siano conformi alle *Norme di Progetto v1.0.0* e rispettino la pianificazione effettuata nel *Piano di Progetto v1.0.0*.

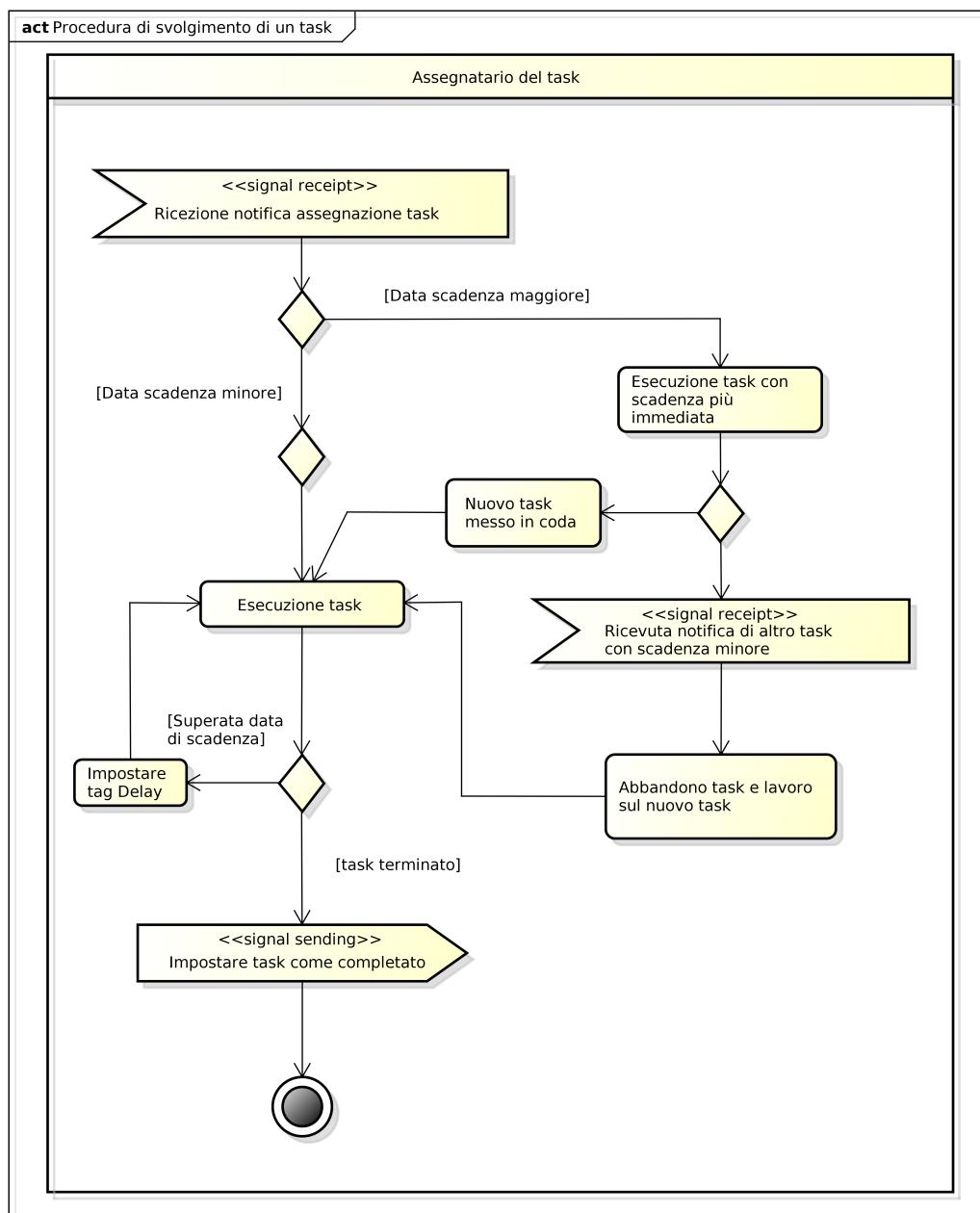


Figura 7: Diagramma di attività - Procedura di svolgimento di un task

4.1.3.1.2 Amministratore di Progetto L' *Amministratore di Progetto* deve svolgere i seguenti compiti:

- Assicurarsi che tutte le risorse siano presenti e operanti;
- Garantire un'infrastruttura funzionale;
- Fornire procedure che servono a garantire la qualità del prodotto uscente da un determinato compito.

4.1.3.1.3 Analista L' *Analista* deve svolgere i seguenti compiti:

- Tradurre il bisogno del cliente in una specifica utile per trovare una soluzione;
- Comprendere la complessità del problema;
- Capire il dominio nel quale lavora il cliente;
- Analizzare il dominio applicativo e le specifiche per poi produrre i documenti di analisi.

4.1.3.1.4 Progettista Il *Progettista* deve svolgere i seguenti compiti:

- Individuare la tecnologia più idonea per risolvere il problema indicato dall'*Analista*;
- Descrivere il funzionamento interno del sistema a diversi livelli di dettaglio;
- Produrre una soluzione comprensibile e attuabile.

4.1.3.1.5 Programmatore Il *Programmatore* ha responsabilità sulle attività di codifica e pertanto deve svolgere i seguenti compiti:

- Scrivere codice documentato, versionato e manutenibile;
- Implementare le soluzioni descritte dal *Progettista*;
- Implementare i test sul codice prodotto.

4.1.3.1.6 Verificatore Il *Verificatore* è il responsabile delle attività di verifica e pertanto deve svolgere i seguenti compiti:

- Controllare che vengano rispettate le norme di progetto;
- Assicurarsi la conformità di ogni stadio del ciclo di vita_G del prodotto.

4.1.4 Strumenti

4.1.4.1 TeXstudio TeXstudio_G è l'*editor* di testo multipiattaforma *open-source*_G utilizzato per scrivere documenti in \LaTeX .

4.1.4.2 Teamwork Teamwork_G è l'applicazione *web* scelta per la gestione dei *task*_G; permette anche di gestire un calendario dove inserire note o fissare appuntamenti e/o traguardi importanti.

4.1.4.3 Astah Astah_G è l'applicativo scelto per la creazione di grafici UML_G. La versione adottata è quella *Professional*, resa disponibile gratuitamente per un utilizzo da parte di studenti.

4.1.4.4 Microsoft Project 2016 Microsoft Project 2016 è il *software* utilizzato per la creazione dei grafici di Gantt_G. Si utilizza il contratto Microsoft MSDN-AA (*Academic Alliance*) reso disponibile dal Dipartimento di Matematica dell'Università degli Studi di Padova in accordo con Microsoft.

4.1.4.5 Telegram Si utilizza Telegram_G per una comunicazione informale all'interno del gruppo. Inoltre Telegram fornisce il vantaggio di essere un'applicazione multipiattaforma_G disponibile nelle seguenti versioni: *desktop*, *web* e *mobile*.

4.1.4.6 Microsoft Office PowerPoint PowerPoint_G è il software utilizzato per creare presentazioni.

4.2 Gestione delle infrastrutture

4.2.0.7 Attività

4.2.0.7.1 Gestione del repository Il gruppo ha deciso di utilizzare un repository_G utile a svolgere funzioni diverse, ma necessarie, allo sviluppo del sistema finale. Una volta iscritto, ciascun membro ha la possibilità di creare il suo *branch*_G personale contenente una copia dei file originali del *branch master* in modo da poter lavorare su delle copie in locale.

4.2.0.7.2 Gestione del messaggio di commit Per mantenere l'ambiente di lavoro il meno ambiguo possibile, è stato deciso di adottare un formato standard per andare a scrivere il messaggio della *commit*_G.

4.2.0.8 Procedure

4.2.0.8.1 Installazione di Git La procedura di installazione varia a seconda del sistema operativo utilizzato. Per i sistemi Linux_G occorre rispettare la seguente procedura:

- Aprire il terminale;
- Immettere il comando *sudo apt-get update*;
- Immettere il comando *apt-get install git*.

Per sistemi OS X_G:

- Recarsi nella sezione dedicata ai *download* <https://git-scm.com/download/mac>;
- Scaricare il *file* in formato DMG_G;
- Aprire il file appena scaricato;
- Lanciare l'installazione cliccando su *git.pkg*.

Infine per i sistemi Windows_G, è necessario fare quanto segue:

- Accedere al sito ufficiale <https://git-for-windows.github.io>;

- Scaricare l'eseguibile;
- Lanciare l'eseguibile;
- Seguire la procedura riportata dalla finestra di dialogo.

4.2.0.8.2 Creazione di una cartella locale di repository Seguire la seguente procedura:

- Creare una nuova cartella;
- Aprire il terminale;
- Collocarsi all'interno della cartella appena creata;
- Eseguire il comando `git init`.
- Immettere il comando `git clone <indirizzo>`, sostituendo `<indirizzo>` con l'URL del progetto su GitHub_G.

4.2.0.8.3 Creazione del branch personale Per creare il *branch_G* personale occorre seguire i seguenti passi:

- Muoversi nella cartella di *repository_G*.
- Accedere alla cartella di progetto;
- Eseguire il comando `git branch <nome>`, sostituendo `<nome>` con il nome del nuovo *branch* da creare.

4.2.1 Norme

4.2.1.1 Repository

4.2.1.1.1 Nomi dei file in SiVoDiM I *file* e le cartelle presenti nel *repository_G* devono essere conformi al seguente formalismo tratto dallo Standard ISO_G 9660:1999 (Level 2):

- I caratteri usati sono solo quelli minuscoli a-z, 0-9, l'*underscore_G* e il punto (esempio: `nome_del_documento.tex`);
- Non sono ammessi caratteri accentati;
- I nomi non possono includere spazi o finire con un punto (.) ;
- I nomi non devono contenere più di un punto (.) utilizzato per separare il nome del *file* dall'estensione (esempio: `studio_di_fattibilita_v1_0_0.pdf`);
- I nomi non devono essere più lunghi di 21 caratteri esclusi i 3 destinati all'estensione.

4.2.1.1.2 Struttura di SiVoDiM Le cartelle nel $repository_G$ verranno organizzate nel seguente modo a partire dalla $root_G$:

- **Documenti:** in essa sono presenti le seguenti cartelle corrispondenti alle varie fasi dello sviluppo:
 - RR: contenente i documenti e i file necessari alla revisione dei requisiti;
 - RP: contenente i documenti e i file necessari alla revisione di progettazione;
 - RQ: contenente i documenti e i file necessari alla revisione di qualifica;
 - RA: contenente i documenti e i file necessari alla revisione dell'accettazione;
- **Codice:** la struttura di questa cartella verrà fornita in una fase successiva del lavoro.

4.2.1.1.3 Messaggio di commit Il messaggio di $commit_G$ dovrà essere conforme alla seguente notazione:

Desc:

Data:

Note:

dove:

- **Desc:** fornisce una descrizione esaustiva dell'attività svolta;
- **Data:** fornisce la data in cui si è apportata la modifica;
- **Note:** aiuta a specificare lo stato del lavoro, nello specifico si adotteranno le seguenti notazioni:
 - C se il lavoro è stato completato;
 - NC se il lavoro non è stato completato;
 - V se il lavoro necessita di verifica;

4.2.2 Strumenti

4.2.2.1 Git Git_G è il sistema di controllo di versione utilizzato per il $repository_G$ del *team*.

4.2.2.2 GitHub $GitHub_G$ è il servizio web di $hosting_G$ adottato per tenere una copia del $repository_G$ del progetto.

4.2.2.3 Dropbox $Dropbox_G$ è lo strumento di $cloud_G$ che si è scelto per gestire *file* che non necessitano di essere sottoposti a controllo di versione.

4.2.2.4 Android Studio Android Studio è l' IDE_G ufficiale per lo sviluppo di applicazioni $Android_G$, basato su IntelliJ IDEA. In aggiunta agli strumenti di sviluppo ereditati da IntelliJ, Android Studio offre il supporto alla compilazione basata su $Gradle_G$, la creazione di interfacce utente basate su $template_G$ e un *editor* integrato che permette di manipolare l' UI_G . Android Studio è disponibile per $Windows_G$, OS X $_G$ e Linux $_G$.

4.2.2.5 Sistema Operativo I membri del gruppo operano su tre diversi sistemi operativi:

- Ubuntu 14.04_G;
- Linux Mint 17_G;
- Windows 10_G;
- OS X 10.11_G.

4.3 Formazione dei membri del gruppo

I membri del gruppo che non hanno conoscenze sufficienti per far fronte ai problemi assegnati dal *Responsabile di Progetto*, dovranno documentarsi e colmare eventuali lacune durante ore esterne a quelle di lavoro, non imputabili perciò ai costi del Committente.