Piano Di Qualifica

Informazioni sul documento

Titolo documento	Piano Di Qualifica
Versione attuale	v2.0.0
Data versione attuale	2012/01/30
Data creazione	2011/12/11
Redazione	Umberto Dall'Est Stefano Faoro Luca Lorenzini Giacomo Lorigiola
Revisione	Giacomo Lorigiola [v2.0.0] Andrea Zironda [v1.0.0]
Approvazione	Umberto Dall'Est
Stato documento	Formale
Uso	Esterno
Distribuito da	SevenFold
Destinato a	SevenFold

Sommario

Il presente documento descrive la strategia di verifica e validazione seguita nello sviluppo del progetto.

Diario delle modifiche

Versione	Data	Autore	Modifiche
v2.0.0	2012/01/30	Umberto Dall'Est	Approvazione e rilascio seconda versione
v1.9.0	2012/01/30	Umberto Dall'Est	Correzione e ampliamento capitolo "Gestione amministrativa della revisione"
v1.8.0	2012/01/29	Umberto Dall'Est	Terminata analisi e stesura di tecniche, strumenti, metodi e test
v1.7.0	2012/01/27	Stefano Faoro	Inserita sezione "Pianificazione delle attività di test"
v1.6.0	2012/01/27	Umberto Dall'Est	Riscritta sezione "Visione generale della strategia di verifica"
v1.5.0	2012/01/27	Umberto Dall'Est	Cambiata parzialmente struttura documento
v1.4.0	2012/01/25	Luca Lorenzini	Aggiunto capitolo "Attività pianificata di test"
v1.3.0	2012/01/24	Luca Lorenzini	Aggiunto capitolo "Resoconto delle verifiche"
v1.2.0	2012/01/17	Stefano Faoro	Aggiunto capitolo "Esito delle revisioni"
v1.1.0	2012/01/16	Stefano Faoro	Aggiunta paragrafi "Analisi statica","Analisi dinamica" e aggiornato paragrafo 2.3.1
v1.0.0	2011/12/16	Antonio Pretto	Approvazione e rilascio prima versione
v0.3.1	2011/12/15	Luca Lorenzini	Migliore definizione del punti 3.1, 3.2 e 3.3
v0.3.0	2011/12/15	Giacomo Lorigiola	Aggiunto "Trattamento discrepanze"
v0.2.1	2011/12/14	Luca Lorenzini	Modificato punto "Comunicazione e risoluzione di anomalie"
v0.2.0	2011/12/12	Luca Lorenzini	Aggiunto punto "Metodi"
v0.1.0	2011/12/11	Luca Lorenzini	Creazione documento

Indice

Elenco delle tabelle

1 Introduzione

1.1 Scopo del documento

Con questo documento si intende garantire e gestire la qualità del prodotto software definendo la strategia generale di verifica e di validazione messa in atto per il progetto PMAC[g]. La prima versione del documento avrà valore contrattuale per la gara d'appalto, poi verrà costantemente aggiornato per coprire ogni necessità nello sviluppo del progetto.

1.2 Scopo del prodotto

Il sistema software PMAC si pone come obiettivo la realizzazione di una piattaforma innovativa per l'apprendimento comportamentale nell'ambito della sicurezza del lavoro, che utilizzi le tecniche della gamification[g] per incentivare il coinvolgimento e la partecipazione degli utenti e per scardinare l'instaurarsi di abitudini errate.

1.3 Glossario

Per evitare ridondanze tutti i termini e gli acronimi presenti nel seguente documento che necessitano di definizione saranno seguiti da una "g" ad apice (E.g. User[g]) alla loro prima occorrenza e saranno riportati in un documento esterno denominato *Glossario.pdf*.
Tale documento accompagna e completa il presente e consiste in un listato ordinato di termini e acronimi con le rispettive spiegazioni.

1.4 Riferimenti

1.4.1 Normativi

- Capitolato d'appalto:
  `http://www.math.unipd.it/~tullio/IS-1/2011/Progetto/C3.pdf`
- Norme generali di progetto:
  vedi documento fornito in allegato *NormeDiProgetto.pdf*
- Analisi dei requisiti:
  vedi documento fornito in allegato *AnalisiDeiRequisiti.pdf*
- Piano di progetto:
  vedi documento fornito in allegato *PianoDiProgetto.pdf*

2 Visione generale della strategia di verifica

2.1 Organizzazione

Ogni modifica effettuata su di un documento, testo o codice che sia, ne cambia lo stato e in particolare ne contesta lo stato ”verificato” provocando lo stanziamento di un nuovo processo di verifica.
Questo processo, a cura di un verificatore, richiederà l'analisi del diario delle modifiche.
Nel caso di una modifica circoscritta il verificatore dovrà controllare solamente la sezione interessata dalla modifica, altrimenti il processo dovrà esaminare ogni sezione influenzata dalla suddetta.
Nell'eventualità che vengano riscontrate anomalie o errori si dovrà procedere con l'impostare lo stato del documento nuovamente a ”instabile” e con la creazione di una segnalazione tramite Ticket.
In caso contrario il documento verrà marcato come ”verificato”.

2.2 Pianificazione strategica e temporale

La pianificazione delle attività di verifica è incaricata al responsabile di progetto il quale deve definire e assegnare tali attività ai verificatori accompagnate dalle relative date di scadenza.
Dovrà inoltre controllarne ed approvarne gli esiti.
Per un esame in dettaglio della distribuzione dei ruoli si veda il documento *Organigramma.pdf*.
Le principali revisioni sono previste nelle seguenti date:

- (RR) Revisione dei Requisiti: 2012/01/10
- (RP) Revisione di Progettazione: 2012/02/06
- (RQ) Revisione di Qualifica: 2012/03/05
- (RA) Revisione di Accettazione: sconosciuta

2.3 Responsabilità

Per un esame in dettaglio dei ruoli e delle relative responsabilità si veda il documento *PianoDiProgetto.pdf*.

2.4 Risorse

Questo paragrafo ha lo scopo di illustrare in breve le risorse richieste dalle attività di verifica.

2.4.1 Risorse necessarie

Nei processi di revisione sono richieste diverse tipologie di risorse.
In primo luogo le risorse umane necessarie per revisionare i documenti.
Seguono le risorse software fra cui si richiedono strumenti per la valutazione delle metriche del codice, strumenti per la verifica del codice Android[g] e strumenti per la verifica del codice RoR[g].

2.4.2 Risorse disponibili

Tra le risorse umane si trovano il verificatore e il responsabile di progetto.
Seguono le risorse software fra cui si trovano Eclipse con alcuni plugin per la valutazione delle metriche del codice Android, la classe AndroidTestCase per i test di unità del codice Android, la gemma "metric_fu" affiancata dal wrapper "metrical" per la valutazione delle metriche del codice RoR, la gemma "rails_best_practises" per verificare l'adesione alle best practises del codice RoR e il framework[g] Rspec per i test di unità del codice RoR.
Infine si richiede l'accesso esclusivo ai documenti da parte dei verificatori durante le fasi di verifica.

2.5 Strumenti

Le risorse software a disposizione dei verificatori sono i seguenti:

- **Apollo**[g]: project manager utilizzato per assegnare le attività di verifica dal responsabile di progetto. Viene inoltre utilizzato dai verificatori per tener traccia dello stato del documento e per la segnalazione di anomalie o errori tramite Ticket.
- **GitHub**[g]: utilizzato per segnalare un errore minore sul codice.
- **TexMaker**: IDE utilizzato per la redazione e la correzione ortografica dei documenti in LaTex[g].
- **Eclipse**: IDE utilizzato nella redazione, nell'analisi delle metriche e nei test di unità del codice Android.
- **Metrics**: plugin per Eclipse votato al calcolo delle metriche del codice.
- **AndroidTestCase**: principale classe dedicata ai test di unità del codice Android.
- **Gemma metric_fu**: gemma votata al calcolo delle metriche del codice RoR.
- **Gemma metrical**: gemma utilizzata per semplificare la gestione di "metric_fu".
- **Gemma rails_best_practises**: gemma che analizza il codice RoR e segnala la violazione delle best practise di RoR.
- **Gemma Rspec**: gemma utilizzata per effettuare i test di unità del codice RoR.
- **W3C**: si utilizzano gli strumenti di validazione del W3C per validare le pagine prodotte dal codice RoR.

2.6 Tecniche

2.6.1 Analisi Statica

L'analisi statica verrà effettuata dai verificatori per tutta la durata del progetto su ogni documento prodotto e si applica con due diverse tecniche di lettura:

- **Walkthrough**: questa tecnica consiste nella lettura critica da parte del verificatore di tutto il contenuto del documento in analisi.
  Risulta essere molto onerosa in termini di tempo e risorse, ma è inevitabile ricorrervi soprattutto nella fase di correzione dei documenti per effettuarne la correzione ortografica e strutturale.
  Ogni errore riscontrato verrà discusso con l'autore del documento per evitare di incorrere in incomprensioni.
  La tecnica di walkthrough sarà molto utilizzata nelle fasi iniziali del ciclo di vita e prima del rilascio di ogni documento.
  Con l'acquisizione di esperienza da parte dei componenti del gruppo attraverso la revisione dei documenti e le discussioni sugli errori più comuni riscontrati durante lo sviluppo del progetto ci si affiderà sempre di più alla tecnica di inspection.
  Il gruppo sarà quindi tenuto a tenere traccia delle problematiche rilevate più spesso durante le fasi di verifica.

- **Inspection**: questa tecnica consiste nella lettura mirata dei documenti incentrata sul controllo dei punti più soggetti ad errori. Durante le fasi di verifica del progetto l'inspection verrà preferita al walthrough in quanto richiede meno risorse.

2.6.2 Analisi Dinamica

Durante lo sviluppo del progetto si seguirà un approccio TDD (Test Driven Development). La progettazione e l'implementazione delle classi di test sarà quindi precedente alla scrittura del codice del sistema e permetteranno lo svolgimento di test ripetibili.
Si prevedono le seguenti tipologie di test:

- **Test di unità**: si occupa della verifica di uno o più moduli che compongono un'unità software. Le statistiche riportano che di media i due terzi degli errori identificati nell'analisi dinamica vengono riscontrati grazie ai test di unità.
  L'esito del test è positivo se l'unità esaminata soddisfa tutti i requisiti per lei previsti.
  I principali strumenti utilizzati per questa tipologia di test saranno AndroidTestCase e Rspec.

- **Test di integrazione**: si occupa della verifica dei sottosistemi che si creano dall'unione di più unità del sistema che dovranno prima aver superato i test di unità per loro previsti. Oltre a venire identificati errori residui sul funzionamento interno delle unità, verranno inoltre controllati i risultati ottenuti dall'integrazione di queste ultime.
  I principali strumenti utilizzati per questa tipologia di test saranno AndroidTestCase e Rspec.

- **Test di sistema**: si occupa della verifica del comportamento del sistema completo rispetto ai requisiti software.
  I principali strumenti utilizzati per questa tipologia di test saranno valutati in seguito.
  Attualmente gli strumenti candidati sono Selenium per testare il codice RoR e i tool monkeyrunner e UI/Application Exerciser Monkey per testare il codice Android.

- **Test di regressione**: si occupano di verificare il corretto funzionamento del sistema al seguito di modifiche.
  Effettuata una modifica verranno quindi rieseguiti i test di unità sulle unità interessate, seguiti dai test di integrazione e al test di sistema.

- **Test di accettazione**: si occupa di sottoporre il sistema al proponente. Se costui riscontra il corretto funzionamento del prodotto e l'adesione a tutti i requisiti allora il sistema verrà rilasciato.

2.6.3 Misurazione delle metriche del codice

Per garantire una stesura lineare del codice ed avere quindi una maggiore leggibilità che aiuti poi la verifica ed il mantenimento del codice, per evitare che la sua complessità aumenti esponenzialmente con l'aggiunta di nuove funzionalità, ma soprattutto per garantire una soddisfacente efficienza prestazionale al nostro sistema, prevediamo la costante analisi delle metriche del codice. Compito del verificatore è quello di verificare che queste metriche rispettino dei valori accettabili. Una descrizione della metrica e un valore di riferimento ritenuto sufficiente per superare la verifica vengono esposti di seguito:

- **Numero di parametri di un metodo**: un numero di parametri formali troppo elevato potrebbe indicare che parte di essi, se correlati, potrebbero essere racchiusi in un'ulteriore classe aumentando la manutenibilità del codice.
  Si cercherà di limitare il numero dei parametri a 10.

- **Conteggio righe di codice di un metodo**: un numero di righe di codice troppo elevato all'interno di un metodo potrebbe indicare che parte delle sue funzionalità potrebbero venir suddivise in ulteriori metodi aumentando così la verificabilità e la manutenibilità del codice.
  Si cercherà di limitare il numero di linee di codice a 1000.

- **Complessità ciclomatica di un metodo**: è il valore che indica la complessità strutturale del codice. Viene definita calcolando il numero di percorsi di codice linearmente indipendenti nel flusso del programma(e.g. blocchi if, cicli for, istruzioni switch, etc.) aggiungendo 1 al totale.
  Un numero troppo elevato riduce la riusabilità e la manutenibilità del codice.
  Si cercherà di limitare l'indice di complessità ciclomatica a 10.

- **Livello di annidamento di un metodo**: un livello di annidamento troppo elevato aumenta la complessità del codice rendendolo difficilmente verificabile e mantenibile.
  Si cercherà di limitare il livello di annidamento a 5.

- **Numero di campi dati di una classe**: un numero di attributi troppo elevato potrebbe indicare che parte di essi, se correlati, potrebbero essere racchiusi in un'ulteriore classe aumentando la verificabilità, la manutenibilità e l'astrazione del codice.
  Si cercherà di limitare il numero dei parametri raggruppandoli dove possibile.

- **Peso di una classe**: è la somma delle complessità ciclomatiche dei metodi contenuti in essa. Un valore troppo elevato potrebbe indicare che parte delle funzionalità offerte dalla classe possano essere delegate ad altre classi.
  Per questa metrica non si prevede un indice di riferimento, ma ci si affiderà alla discrezione del verificatore nel ritenere questo indice non accettabile.

- **Grado di accoppiamento di una classe**: rappresenta la quantità di dipendenze esterne richieste per il corretto funzionamento di una classe.
  Seppure non si calcolerà e analizzerà questo indice il codice verrà redatto cercando di mantenere un basso grado di accoppiamento fra classi e package in modo da rendere il codice più indipendente, modulare, verificabile e mantenibile possibile.

2.7 Metodi

2.7.1 Ticketing

La procedura di segnalazione degli errori e dell'assegnazione delle attività di correzione è denominata ticketing.
Con i ticket un verificatore assegna la correzione di un errore riscontrato ad un progettista o a un programmatore che deve seguire le norme descritte al capitolo 4.2 del documento *NormeDiProgetto.pdf*
La creazione di un ticket può comunque essere eseguita indipendentemente dal ruolo che si ricopre, previa identificazione di un errore. In caso un ticket sia creato in modo errato si deve procedere contattando il creatore del ticket chiedendo delucidazioni a riguardo per evitare di perdere l'utilità dello stesso.
In generale la segnalazione di un errore non deve contenere indicazioni sull'entità della correzione, ad eccezione di anomalie la cui spiegazione risulterebbe particolarmente complicata se priva di essa.
La procedura di creazione dei ticket prevede che per il riscontro di più errori in un'unica sessione di verifica vengano aperti più ticket per avere un resoconto e uno storico più dettagliato.

3 Gestione amministrativa della revisione

3.1 Comunicazione e risoluzione di anomalie

Un'anomalia si presenta quando in seguito ad una revisione si riscontrano degli errori di programmazione o di stesura dei documenti che non riguardano la progettazione, ma sono nati nelle fasi successive.
Nel caso si trovi un'anomalia se ne deve comunicare la presenza al redattore del documento utilizzando il ticketing. Per maggiori informazioni sul metodo del ticketing consultare l'apposito paragrafo sezione metodi di questo documento.
Le varie tipologie di anomalie sono:

1. **Anomalie di contenuto:** un'anomalia di contenuto è uno scostamento dei contenuti di un documento da quelli previsti per lo stesso dall'analista e dal progettista.

2. **Anomalia formale:** un'anomalia formale è identificata ogni volta che si incontra un errore sintattico od ortografico nei documenti.

3. **Anomalie di codice:** un'anomalia di codice consiste in un errore logico o sintattico commesso nel codice.

4. **Anomalia strutturale:** un'anomalia strutturale consiste in un errore riscontrato nella struttura di un documento.

3.2 Trattamento discrepanze

Sarà considerata discrepanza uno scostamento del prodotto dalle attese del cliente e dello stesso gruppo SevenFold in base ai requisiti specificati nell'*AnalisiDeiRequisiti.pdf* e quindi dal prodotto atteso. Solitamente una discrepanza è attribuibile ad un errore di analisi o di progettazione.
A seguito del riscontro di una discrepanza si procederà con l'apertura di un apposito ticket che descriverà la discrepanza in dettaglio e che sarà assegnato al responsabile del progetto il quale avrà il compito di decidere sul da farsi.
Di norma tali discrepanze verranno riscontrate dai verificatori durante la loro normale attività di verifica, ma potranno essere sollevate anche da una qualsiasi figura del gruppo nel momento in cui questa si accorga della loro presenza all'interno del prodotto.
Il responsabile, analizzata la discrepanza, dovrà assegnare alla stessa un livello di gravità valutato sulla base dell'importanza del scostamento dalle attese, e sulla base dei costi e dei tempi necessari per risolverla.
Verrà quindi definito un piano correttivo che analizzerà in dettaglio come procedere e si procederà con l'assegnazione delle attività di correzione e successivamente con quelle di verifica.
A discrepanza risolta verrà effettuato un test di regressione per verificare la perfetta compatibilità delle modifiche apportate con il resto del sistema.
A test concluso, verrà chiuso il ticket di notifica della discrepanza.

3.3 Procedure di controllo di qualità di processo

Per assicurare la qualità di processo si delega al ruolo di responsabile di progetto la misurazione dei tempi, delle risorse e dei costi richiesti al completamento di un processo.
Supervisionando costantemente tutti i membri del gruppo di lavoro e i documenti da loro prodotti il responsabile monitora potenziali scostamenti dalle norme previste per il processo corrente e può intervenire riprendendo il diretto interessato dell'infrazione.
Il miglioramento continuo della qualità di processo è garantito dal tracciamento di ogni considerazione a suo riguardo da parte del responsabile che maturerà quindi nel tempo una maggiore esperienza a riguardo che potrà poi applicare alla successiva iterazione del modello incrementale.

4 Pianificazione delle attività di test

4.1 Test di Sistema

Con i test di sistema si andrà a verificare che tutti i requisiti funzionali presenti nel documento *AnalisiDeiRequisiti.pdf* siano stati effettivamente implementati nel sistema prima del rilascio finale.
Ogni requisito sarà testato con una prova dinamica utilizzando le modalità e gli strumenti descritti per i test di sistema nel paragrafo 2.6.2 "Analisi Dinamica".
La notazione usata per i codici delle prove consta del codice del requisito preceduto da una "T".

Tabella 4.1: Tracciamento requisito - verifica

Requisito	Codice prova	Descrizione prova
Fo-1	TFo-1	Si verifica che il sistema inserisca correttamente il workgroup creato dal PMAC Administrator[g].
Fo-2.1.1	TFo-2.1.1	Si verifica che il sistema inserisca correttamente l'achievement[g] creato dal PMAC Administrator.
Fo-2.1.1	TFo-2.1.1	Si verifica che il sistema inserisca correttamente il badge[g] creato dal PMAC Administrator.
Fo-2.2	TFo-2.2	Si verifica che il sistema inserisca correttamente il nuovo profilo aziendale creato dal PMAC Administrator, salvando tutti i campi del profilo descritti dai sotto-requisiti di Fo-2.2 nell'Analisi dei Requisiti.
Fo-2.3	TFo-2.3	Si verifica che il sistema abbia salvato tutte le modifiche effettuate ai campi del profilo aziendale da parte del PMAC Administrator.
Fo-2.4	TFo-2.4	Si verifica che il sistema elimini correttamente il profilo aziendale selezionato dal PMAC Administrator.
Fo-2.5	TFo-2.5	Si verifica che all'invio di un ticket da parte di un Super-user[g] il sistema riceva effettivamente il ticket destinato al PMAC Administrator.
Fo-3.1	TFo-3.1	Si verifica che il sistema inserisca correttamente la nuova quest[g] creata dal PMAC Administrator, salvando tutti i dati della stessa descritti dai sotto-requisiti di Fo-3.1 nell'Analisi dei Requisiti. Nell'assenza di un campo il sistema deve rifiutare l'inserimento della quest e riproporre la schermata di inserimento.
Fo-3.2	TFo-3.2	Si verifica che il sistema associ correttamente una quest al workgroup selezionato dal PMAC Administrator.

(Continua alla pagina successiva)

(Continua dalla pagina precedente)

Fo-3.3	TFo-3.3	Si verifica che il sistema elimini correttamente la quest specificata dal PMAC Administrator.
Fo-4.1	TFo-4.1	Si verifica che il sistema inserisca correttamente il nuovo User registrato dal Super-user salvando tutti i dati dello stesso descritti dai sotto-requisiti di Fo-3.1 nell'Analisi dei Requisiti. Il sistema dovrà rispondere in maniera negativa nel caso in cui l'indirizzo e-mail del nuovo User sia già occupata da un altro User.
Fo-4.2	TFo-4.2	Si verifica che il sistema abbia salvato tutte le modifiche effettuate ai campi del profilo dello User specificato dal Super-user.
Fo-4.3	TFo-4.3	Si verifica che il sistema elimini correttamente lo User selezionato dal Super-user.
Fo-4.4	TFo-4.4	Si verifica che il sistema visualizzi correttamente i profili e le statistiche dello User selezionato dal Super-user.
Fo-4.6	TFo-4.6	Si verifica che il sistema esegua correttamente la creazione del ticket da parte del Super-user e che lo invii correttamente.
Fo-5	TFo-5	Si verifica che il sistema permetta la risoluzione di tutte le tipologie di quest dedicate all'Desktop-user[g].
Fo-6	TFo-6	Si verifica che il sistema permetta la risoluzione di tutte le tipologie di quest al Mobile-user[g].
Fo-7	TFo-7	Si verifica che il sistema permetta l'autenticazione di uno User, rispondendo negativamente nel caso in cui lo User inserisca indirizzo e-mail o password errati.
Fo-8	TFo-8	Si verifica che il sistema notifichi a video la presenza di nuove quest inviate allo User.
Fo-8.1	TFo-8.1	Si verifica che il sistema di notifica permetta il collegamento diretto alla pagina di risoluzione delle quest ricevute dallo User.
Fo-9	TFo-9	Si verifica che il sistema permetta la visualizzazione delle classifiche aggiornate a seguito della richiesta da parte di uno User.
Fo-10.1	TFo-10.1	Si verifica che il sistema abbia salvato correttamente tutte le modifiche effettuate dallo User al proprio profilo.
Fo-10.2	TFo-10.2	Si verifica che il sistema permetta la visualizzazione del profilo personale a seguito della richiesta da parte di uno User.
Fo-10.3	TFo-10.3	Si verifica che il sistema permetta la visualizzazione delle statistiche dello User che le richiede, visualizzando tutti i dati descritti dai sotto-requisiti di Fo-10.3 nell'Analisi dei Requisiti.
Fd-1	TFd-1	Si verifica che il sistema permetta allo User la visualizzazione dei profili e delle statistiche di un qualsiasi User selezionato.
Fd-2	TFd-2	Si testa che il sistema visualizzi la breve nota esplicativa dopo la risoluzione di una quest.

5 Resoconto delle attività di verifica

5.1 Dettaglio delle verifiche tramite analisi

5.1.1 Revisione dei requisiti

Durante questa fase iniziale si è fatto largo utilizzo della tecnica del walkthrough prestando particolare attenzione a:

- Controllare la correttezza ortografica utilizzando TexMaker
- Controllare la correttezza sintattica del documento
- Controllare la correttezza dei diagrammi, delle immagini e dei contenuti del documento
- Controllare che il redattore del documento abbia effettivamente seguito le norme definite dal documento *NormeDiProgetto.pdf*

Per ogni modifica apportata in seguito alla revisione del documento si è fatto uso della tecnica dell'inspection.
Prima del rilascio del documento è stata effettuata un'ultima rilettura completa del documento. Ogni documento in uscita dalla fase RR è stato verificato, corretto secondo le modalità previste e le modifiche/correzioni effettuate sono state riportate nel "Diario delle modifiche" inserendo l'autore della modifica, la data, il soggetto della modifica e la versione del documento.

5.1.2 Revisione di progettazione

Come nella fase precedente si è fatto largo uso delle tecniche di analisi statica e dei suoi strumenti.
La maggiore esperienza maturata nella fase precedente ci ha permesso di fare maggiore uso della tecnica dell'inspection preferendola al walkthrough dove possibile per ridurre l'impiego delle risorse.
Buona parte del tempo è stata utilizzata per la correzione e l'ampliamento dei documenti della fase precedente. Per maggiori dettagli consultare "Dettaglio dell'esito delle revisioni".

5.2 Dettaglio delle verifiche tramite prove

Il resoconto dei test verrà stilato non appena saranno eseguite attività di verifica che richiedano la tecnica dell'analisi dinamica.

5.3 Dettaglio dell'esito delle revisioni

Ad ogni revisione il gruppo riceverà da parte del committente un esito riguardante la documentazione consegnata, che indicherà il grado di correttezza del lavoro svolto dal gruppo fino a quel momento.
Di seguito verranno descritte e le principali attività di verifica e correzione svolte dal gruppo a seguito delle revisioni.

5.3.1 Revisione dei Requisiti

A seguito della Revisione dei Requisiti sono stati riesaminati e corretti i seguenti documenti:

- **Analisi dei Requisiti**: sono stati modificati i paragrafi "Contesto d'uso" e "Funzionalità del prodotto" cercando di definirne meglio i contenuti, inoltre è stato rimosso il paragrafo "Assunzioni e dipendenze".
  Sono stati corretti e revisionati gran parte dei diagrammi degli Use Case[g], aggiungendone anche di nuovi per aumentarne il livello di dettaglio.
  Sono stati rivisitati di conseguenza i relativi requisiti e sono stati modificati i requisiti segnalati come ambigui in maniera da renderli più specifici e verificabili durante il corso del progetto.

- **Piano di Progetto**: aggiustato in minima parte il ciclo di vita per ovviare ai ritardi maturati durante la fase di revisione dei requisiti e aggiustate di conseguenza alcune sezioni del documento. Per maggiori dettagli consultare il capitolo 2 del documento *PianoDiProgetto.pdf*.

- **Piano di Qualifica**: dopo aver aggiustato la struttura del documento sono stati aggiunti dettagli riguardanti le tecniche di analisi, le attività di verifica effettuata nella prima fase e i resoconti delle attività di verifica per le fasi Revisione dei Requisiti e Revisione di progettazione.
  Sono inoltre stati aggiunti dettagli sulle metriche del codice e, dopo un'approfondita ricerca, anche gli strumenti che si sono scelti per tutte le future verifiche sul codice.

6 Pianificazione ed esecuzione del collaudo

Il collaudo per essere significativo deve seguire una procedura standard così da permettere una corretta analisi dei risultati prodotti.

6.1 Specifica della campagna di validazione

La campagna di validazione sarà specificata in seguito all'ultima iterazione e a prodotto ultimato dando modo al proponente di farsi un'idea chiara del sistema e delle modalità più consone richieste al collaudo per offrire un'analisi approfondita di ogni possibile stato del sistema.

6.2 Dettaglio dell'esito della campagna di validazione

Questa sezione sarà redatta una volta svolta la campagna di validazione.