



TEAM COMMITTED

UNIVERSITÀ DEGLI STUDI DI PADOVA

Piano di Qualifica V2.0

Informazioni sul documento

Nome documento	Piano di Qualifica
Data documento	2012/01/23
Redattori	<ul style="list-style-type: none">• Giacomo Quadrio• Lorenzo Braghetto• Gabriele Facchin• Massimo Dalla Pietà
Verificatori	<ul style="list-style-type: none">• Massimo Dalla Pietà• Gabriele Facchin
Approvazione	Giorgio Maggiolo
Uso documento	Esterno <ul style="list-style-type: none">• <i>Team Committed</i>
Lista distribuzione	<ul style="list-style-type: none">• <i>Prof. Tullio Vardanega</i>• <i>Dr. Amir Baldissera</i>

Sommario

Questo documento vuol definire le norme volte alla regolamentazione delle fasi di verifica che il gruppo *Team Committed* ha deciso di adottare.



Diario delle modifiche

Modifica	Autore	Data	Versione
<i>Approvato.</i>	Giorgio Maggiolo	2012/01/23	V2.0
<i>Verificato. In attesa di approvazione</i>	Gabriele Facchin	2012/01/21	V1.8
<i>Inserito capitolo 5: Pianificazione ed esecuzione del collaudo - Test di sistema</i>	Giacomo Quadrio	2012/01/20	V1.7
<i>Inserita sezione 7.2: Revisione di progettazione</i>	Lorenzo Braghetto	2012/01/20	V1.6
<i>Inserito capitolo 5: Pianificazione ed esecuzione del collaudo - Test di integrazione</i>	Lorenzo Braghetto	2012/01/19	V1.5
<i>Inserito capitolo 2.5.3: Strategie per la verifica</i>	Giacomo Quadrio	2012/01/18	V1.4
<i>Ampliate tecniche inspection del codice</i>	Lorenzo Braghetto	2012/01/17	V1.3
<i>Ampliate tecniche inspection dei documenti</i>	Lorenzo Braghetto	2012/01/17	V1.2
<i>Ampliato e corretto capitolo 5.1</i>	Giacomo Quadrio	2012/01/16	V1.1
<i>Approvato.</i>	Giorgio Maggiolo	2011/12/16	V1.0
<i>Piccole modifiche, adeguato alle Norme di Progetto. Revisione completata.</i>	Massimo Dalla Pietá	2011/12/16	V0.11
<i>Completato Resoconto attività di verifica capitolo 5</i>	Massimo Dalla Pietá	2011/12/15	V0.10
<i>Completato Gestione amministrativa della revisione capitolo 4</i>	Massimo Dalla Pietá	2011/12/14	V0.9
<i>Completato Obbiettivi di qualità capitolo 3</i>	Gabriele Facchin	2011/12/14	V0.8
<i>Completato Obbiettivi di qualità capitolo 3, sottocapitolo 3.1</i>	Gabriele Facchin	2011/12/13	V0.7
<i>Completato Visione generale delle strategie di verifica capitolo 2</i>	Gabriele Facchin	2011/12/13	V0.6
<i>Completato Strumenti e Tecniche capitolo 2 sottocapitolo 2.5</i>	Lorenzo Braghetto	2011/12/12	V0.5
<i>Completato Organizzazione capitolo 2 sottocapitolo 2.1</i>	Giacomo Quadrio	2011/12/10	V0.4
<i>Completato Risorse capitolo 2 sottocapitolo 2.4</i>	Lorenzo Braghetto	2011/12/10	V0.4
<i>Iniziata stesura Gestione amministrativa della revisione capitolo 4</i>	Lorenzo Braghetto	2011/12/09	V0.3
<i>Inizio stesura Visione generale delle strategie di verifica capitolo 2</i>	Giacomo Quadrio	2011/12/09	V0.2
<i>Inizio creazione del documento. Creato lo schema base e completata stesura - Introduzione capitolo 1</i>	Quadrio Giacomo	2011/12/09	V0.1

Indice

1	Introduzione	6
1.1	Scopo del documento	6
1.2	Scopo del prodotto	6
1.3	Glossario	6
1.4	Riferimenti	6
1.4.1	Normativi	6
1.4.2	Informativi	6
2	Visione generale delle strategie di verifica	8
2.1	Organizzazione	8
2.2	Pianificazione strategica e temporale	9
2.3	Responsabilità	9
2.4	Risorse	9
2.4.1	Risorse Necessarie	9
2.4.2	Risorse Disponibili	10
2.5	Strumenti, tecniche e metodi	11
2.5.1	Strumenti	11
2.5.2	Tecniche	12
2.5.3	Strategie per la verifica	14
2.5.4	Misure e Metriche	15
3	Obbiettivi di qualità	17
3.1	Qualità dei processi	17
3.2	Qualità del prodotto	19
3.2.1	Funzionalità	20
3.2.2	Affidabilità	20
3.2.3	Efficienza	21
3.2.4	Usabilità	21
3.2.5	Manutenibilità	22
3.2.6	Portabilità	22
4	Gestione amministrativa della revisione	23
4.1	Comunicazione e risoluzione di anomalie	23
4.2	Trattamento delle discrepanze	23
4.3	Procedure di controllo di qualità di processo	24
5	Pianificazione ed esecuzione del collaudo	25
5.1	Test di Sistema	25
5.1.1	Ambito Generale	25
5.1.2	Ambito Utente	26
5.1.3	Ambito Dipendente	27
5.1.4	Ambito Amministratore Azienda	28
5.1.5	Ambito Amministratore Sicurezza	30
5.1.6	Ambito Amministratore Installatore	31

5.2	Test di integrazione	32
5.2.1	Test intra-componenti	33
5.2.2	Test inter-componenti	35
6	Dettaglio dell'esito delle revisioni	36
6.1	Revisione dei Requisiti	36
7	Resoconto attività di verifica	37
7.1	Revisione dei Requisiti	37
7.2	Revisione di Progettazione	37

1 Introduzione

1.1 Scopo del documento

Tale documento ha lo scopo di illustrare le strategie adottate per la verifica e validazione del lavoro svolto dal *Team Committed* relativamente al progetto SafetyGame e ai processi relativi alla sua produzione.

1.2 Scopo del prodotto

Il prodotto SafetyGame si propone di offrire una piattaforma alle aziende per poter rendere alcuni contesti critici come l'apprendimento di norme relative alla sicurezza sul lavoro, parte integrante delle attività lavorative standard. Ad oggi infatti, le tematiche sono affrontate tramite attività di formazione statiche, basate su insegnamenti in aula che poco motivano ed interessano i partecipanti. SafetyGame si prefigge quindi di aumentare l'interesse dei dipendenti di un'azienda tramite l'utilizzo di sfide e compiti da portare a termine che si andranno ad integrare con le attività lavorative reali. Tale scopo si raggiungerà tramite la somministrazione di sfide attraverso apposite applicazioni desktop, browser e mobile.

1.3 Glossario

Tutti i termini marcati da apposita sottolineatura saranno approfonditi e descritti all'interno del file Glossario-v2.0.pdf al fine di evitare ambiguità.

1.4 Riferimenti

1.4.1 Normativi

- **Capitolato d'appalto:** Piattaforma mobile di apprendimento comportamentale, rilasciato dal proponente Mentis s.r.l. reperibile all'indirizzo <http://www.math.unipd.it/~tullio/IS-1/2011/Progetto/C3.pdf>
- **Analisi dei requisiti** (Allegato Analisi_Dei_Requisiti_v2.0.pdf)
- **Norme di progetto** (Allegato Norme_Di_Progetto_v2.0.pdf)
- **Verbale incontro proponente 2011/12/05** (Allegato Verbale05.12.2011.pdf)
- **Verbale incontro proponente 2012/01/26** (Allegato Verbale20120126.pdf)

1.4.2 Informativi

- **ISO/IEC 15504:1998** (www2.cnipa.gov.it/site/_contentfiles/00310300/310320_15504.pdf)
- **ISO/IEC 9126** (http://it.wikipedia.org/wiki/ISO/IEC_9126)

- **ISO 14598** (http://webstore.iec.ch/preview/info_isoiec14598-1%7Bed1.0%7Den.pdf)

2 Visione generale delle strategie di verifica

2.1 Organizzazione

Per poter effettuare un corretto processo di verifica si è scelto di effettuare le dovute operazioni di controllo quando il prodotto sotto analisi avrà raggiunto uno stadio tale per cui risulta presentare sostanziali modifiche rispetto alla precedente versione. Lo svolgimento di tale fase sarà agevolato grazie all'apposito registro delle modifiche. Inoltre, per garantire una migliore verifica si è adottato il “**Broken Window Theory**” secondo il quale, non appena un errore viene rilevato, questo andrà segnalato e corretto il prima possibile onde evitarne la propagazione incontrollata.

Il ciclo di vita scelto per lo sviluppo del progetto è un ciclo di vita incrementale (vedi *Piano di Progetto - V2.0* per approfondimenti) e di conseguenza le operazioni di verifica verranno realizzate in modo tale da intervenire in maniera coerente nelle varie fasi del progetto come illustrato di seguito:

- **Analisi dei requisiti:** ogni documento necessario alla RR, quando verrà completato, entrerà nella dedicata fase di revisione per controllare la presenza di eventuali irregolarità lessico/grammaticali e nei contenuti esposti. Nel dettaglio, il controllo ortografico verrà effettuato con gli strumenti messi a disposizione da LyX, per la precisione tramite il plugin Aspell, mentre il controllo lessicale, grammaticale e sintattico da un'accurata rilettura del testo. I contenuti verranno invece controllati in modo tale da verificare la copertura delle richieste del proponente e questo tramite un'accurata rilettura e confronto con il capitolato d'appalto ed i verbali degli incontri. Si constaterà poi che ogni caso d'uso abbia almeno un requisito corrispondente, effettuando un controllo delle apposite tabelle di tracciamento. Verranno verificati inoltre i contenuti grafici e tabellari e la conformità dei documenti alle Norme di Progetto stabilite. Se durante la verifica saranno state rilevate irregolarità queste verranno segnalate tramite un apposito ticket dal verificatore e corrette dal redattore. Il processo di verifica si concluderà quindi con la validazione del documento da parte del verificatore e l'approvazione da parte del responsabile per la presentazione al committente
- **Progettazione:** il processo di verifica riguardante la fase di Progettazione consisterà nel verificare che tutti i requisiti descritti durante la fase di Analisi dei Requisiti rientrino nei componenti individuati; ciò verrà effettuato tramite un'accurata analisi delle tabelle di tracciamento. Qualora dalla verifica sorgano incongruenze o mancanze, queste verranno segnalate tramite ticket e successivamente risolte
- **Realizzazione:** la verifica in questa fase verrà effettuata sia da parte dei programmatori stessi che utilizzando appositi e specifici strumenti di verifica automatizzata del codice, come elencato nella sezione 2.5.1 dove si possono visionare quelli adottati da Team Committed con una relativa

descrizione della loro funzione. La presenza di errori verrà segnalata da un apposito Ticket che verrà preso in carico dai programmatori e chiuso una volta risolto il problema

- **Validazione:** il team si impegna a fornire al collaudo una versione funzionante e possibilmente completa del prodotto. Eventuali difetti o incongruenze saranno corretti ed eliminati da Team Committed

2.2 Pianificazione strategica e temporale

La strategia di verifica adottata prevede, come scritto in sezione 2.1, una verifica ogni qualvolta il prodotto da noi realizzato abbia raggiunto uno stadio tale che si differenzi dalla sua precedente versione. Questa, unita alla tecnica “*Broken Window Theory*”, cerca di garantire un alto livello di qualità al fine di raggiungere il totale soddisfacimento dei requisiti richiesti dal capitolato d'appalto oltre che del cliente. Il *Responsabile di Progetto* avrà quindi il compito di coordinare e definire le attività volte alla verifica del materiale prodotto, sia esso software, documenti o materiale d'altro genere, oltre che a far rispettare le scadenze previste.

2.3 Responsabilità

Al *Responsabile di Progetto* saranno affidate le responsabilità riguardanti tutte le attività di verifica e validazione, ponendosi quindi come garante del corretto svolgimento delle attività volte alla qualità del materiale prodotto nei confronti del committente. L'*Amministratore di Progetto* si occuperà invece di assicurare che l'ambiente in cui tutte le attività di realizzazione del prodotto si svolgeranno sia adeguato a tale scopo.

2.4 Risorse

La gestione della qualità prevede l'utilizzo di alcune risorse

2.4.1 Risorse Necessarie

Risorse Umane

I ruoli necessari per garantire un'adeguata qualità sono i seguenti:

- **Responsabile:** è il referente e responsabile nei confronti del committente, supervisiona i processi interni e si preoccupa di valutare le proposte di modifica correttiva o migliorativa dei Verificatori
- **Amministratore:** definisce i piani per la gestione della qualità, come i processi di verifica, test e individuazione e risoluzione delle anomalie e discrepanze

- **Verificatore:** esegue le attività di verifica sul prodotto a seconda delle norme redatte, ne riassumerà gli esiti e in caso di discrepanze presenterà il problema al Responsabile
- **Programmatore:** è incentivato a eseguire attività di debugging quando ne riterrà necessario ed è coinvolto dai verificatori nella risoluzione dei ticket

Risorse Software

- **Software** per la verifica dei requisiti richiesti
- **Software** di gestione e commento di ticket, codice e documentazione
- **Software** necessario alla corretta configurazione dell'ambiente di sviluppo
- **Software** utile alla comunicazione fra i membri del team

Risorse Hardware

- **Computer** su cui eseguire il software necessario all'ambiente di sviluppo
- **Luogo** fisico per incontri fra i membri del team

2.4.2 Risorse Disponibili

Risorse Software

- **Github e correlati:** risorse messe a disposizione dal servizio web Github per gestire commenti o problemi a codice, documentazione e ticket
- **Aspell** per il controllo ortografico dei documenti
- **Eclipse** con relativi strumenti già inclusi o aggiunti tramite plugin
- **Strumenti W3C** per validazione web
- **Gruppo Facebook** per poter comunicare fra membri del team

Risorse Hardware

- **Computer** personali, portatili e fissi
- **Computer** messi a disposizione nei laboratori dal Dipartimento di Matematica Pura ed Applicata dell'Università di Padova
- **Aule** studio del Dipartimento di Matematica Pura ed Applicata dell'Università di Padova

2.5 Strumenti, tecniche e metodi

2.5.1 Strumenti

Il gruppo potrà avvalersi dei seguenti strumenti, quelli che seguiranno saranno gli strumenti utili alle fasi di verifica e vengono presi in considerazione per la loro utilità in questa fase.

- **Aspell** ($\geq 0.60.6$): strumento per la correzione grammaticale dei documenti redatti. L'utilizzo di Aspell avverrà tramite apposito plugin per LYX (<http://aspell.net/>)
- **Eclipse** ($\geq 3.7.1$): IDE multi-linguaggio e multi-piattaforma che mette a disposizione alcune funzionalità di debugging quali l'esecuzione step-by-step, l'impostazione di beaking point ecc (<http://www.eclipse.org/>)
- **FindBugs** ($\geq 2.0.0$): strumento utilizzato nell'analisi statica e volto ad individuare errori nella scrittura di codice Java (<http://findbugs.sourceforge.net/>)
- **EclEmma** ($\geq 1.5.3$): strumento in grado di determinare la copertura del codice Java prodotto sia durante le fasi di esecuzione che di testing e disponibile sotto forma di plugin per *Eclipse* (<http://www.eclemma.org/>)
- **Android Lint** ($\geq 16.0.0$): strumento utilizzato per individuare i principali errori di programmazione all'interno di applicazioni Android (<http://tools.android.com/tips/lint>)
- **Metrics** ($\geq 1.3.6$): plugin per *Eclipse* che permette di effettuare analisi metrica del codice prodotto. Fornisce informazioni relative a misure statiche del codice:
 - Complessità ciclomatica
 - Peso delle classi
 - Numero di parametri
 - Numero di campi dati per classe
 - Numero livelli di annidamento
 - Indice di utilità - Indice di dipendenza(<http://metrics.sourceforge.net>)
- **JUnit** (≥ 4.10): Framework utilizzato per effettuare test di unità per il linguaggio Java (<http://www.junit.org/>)
- **Selenium IDE** ($\geq 1.4.1$): plugin per *Firefox* utilizzato per registrare ed eseguire test tramite browser (<http://seleniumhq.org/>)
- **ApacheBench** (≥ 2.0): strumento a linea di comando utilizzato per misurare l'efficienza di un server web ed in grado di simulare situazioni di sovraccarico della rete (<http://www.apache.org/>)

- **SpeedTracer** (≥ 2.4): plugin per *Google Chrome* che permette di verificare l'efficienza di un'applicazione web durante la sua esecuzione (<http://code.google.com/intl/it-IT/webtoolkit/speedtracer/>)
- **Strumenti di validazione W3C:**
 - **Markup Validation Service:** per effettuare test sulle pagine di cui l'applicativo web è costituita e verificarne l'aderenza agli standard HTML5 (<http://validator.w3.org/>)
 - **CSS Validation Service:** per effettuare test sui fogli di stile utilizzati nell'applicativo web e verificarne l'aderenza allo standard CSS 2.1 (<http://jigsaw.w3.org/css-validator/>)

2.5.2 Tecniche

Analisi statica L'analisi statica consisterà nella verifica critica del codice, controllandone requisiti e la corretta progettazione. Ogni attività svolta verrà registrata e documentata.

Questo tipo di analisi è usualmente fatta con due tecniche complementari:

- **Walkthrough:** consiste in un'ispezione generale del codice o dei documenti senza prerequisiti iniziali e pianificazione. È utile nelle fasi iniziali di verifica quando va considerato il software nel suo intero senza sapere quali saranno le parti più critiche. Ogni problema rilevato verrà discusso fra i verificatori e gli sviluppatori (tutti quelli che hanno realizzato la parte di prodotto in questione)
- **Inspection:** consiste in una verifica mirata del codice o dei documenti, è possibile solo quando si è raggiunto un buon grado di conoscenze e si è già consapevoli degli errori più comuni o delle aree più critiche. Riguardo l'Inspection dei documenti seguiremo questi punti:
 - **Latex:**
 - * Impostazione generale del documento affinché aderisca alle Norme di Progetto
 - * Controllo della sillabazione automatica
 - * Controllo punteggiatura e leggibilità del testo
 - * Presenza di una e una sola spaziatura dopo la punteggiatura
 - * Posizionamento corretto delle immagini all'interno della pagina
 - * Accentazioni corrette (scelta corretta tra acuti e gravi)
 - * Lettere maiuscole accentate
 - * Termini definiti nel glossario ma non sottolineati nel documento
 - * Termini sottolineati nel documento ma non presenti nel glossario
 - * Sottolineatura di tutti i termini che sono stati sottolineati almeno una volta all'interno del documento

– UML:

- * Mantenimento della consistenza tra la nomenclatura dei grafici UML e la nomenclatura adottata nelle sezioni in cui i grafici saranno inseriti
- * Controllo ortografico dettagliato, data l'impossibilità di automatizzare tali controlli sui grafici
- * Attenzione alla direzione delle frecce nei diagrammi
- * Attenzione alla perfetta orizzontalità o verticalità di frecce orizzontali o verticali
- * Controllo del nome del sistema
- * Posizione degli include e degli extend
- * Eliminazione di caratteri extra, non previsti dalle norme, creati da Bouml

Per quanto riguarda invece l'inspection del codice, seguiremo alcuni punti guida:

- Le variabili di programma devono essere inizializzate prima che il loro valore sia utilizzato.
- Tutte le costanti devono avere un nome.
- Il limite superiore degli array deve essere uguale alla loro dimensione o alla dimensione-1.
- Controllare la presenza di eventuali buffer overflow.
- Verificare che la condizione delle istruzioni condizionali sia corretta.
- Ogni ciclo deve essere ultimato.
- Le istruzioni composte devono essere correttamente messe fra parentesi.
- Le variabili di input devono essere tutte utilizzate.
- I break devono essere correttamente utilizzati se richiesti.
- Le variabili di input devono essere tutte utilizzate.
- Le variabili di output devono possedere un valore prima che vengano restituite.
- Verificare gli effetti degli input imprevisti.
- Le chiamate a funzione e a metodo devono possedere il corretto numero di parametri.
- Il tipo dei parametri formali e reali deve corrispondere.
- Corretta disposizione dei parametri.
- Verificare che siano state prese in considerazione tutte le possibili condizioni di errore.

Chiaramente la tecnica dell'*inspection* è più efficace, ma sarà possibile attuarla solo dopo aver appreso la consapevolezza iniziale del proprio codice e delle proprie capacità.

Metodi di Analisi Statica:

- **Analisi del flusso di controllo:** si accerta che il codice segua il flusso aspettato, che non si possa entrare in porzioni di codice che possano non terminare, che non esista codice non raggiungibile
 - **Analisi del flusso dei dati:** si accerta che il software non acceda mai a variabili non inizializzate o scriva inutilmente più volte prima di usare una variabile
 - **Analisi del flusso di informazione:** verifica che gli input e gli output di ogni unità di codice o di più unità rientrino nelle specifiche del programma

Analisi dinamica L'analisi dinamica consisterà nella verifica dei componenti del software o del sistema in generale. Verranno effettuati test mirati e ripetuti in diversi contesti. Il programmatore, durante i test, dovrà essere cosciente dello stato dell'ambiente, degli input e dei risultati in ogni momento dell'esecuzione. Questi risultati saranno utili solo nel caso vengano trovati errori da correggere, nel caso non vengano trovati, non significa necessariamente che non ve ne siano.

Metodi di Analisi Dinamica

- **Test di unità:** test che si effettuano per ogni unità del software con il massimo grado di parallelismo
- **Test di integrazione:** verifica dei componenti formati dall'integrazione delle varie unità che hanno passato il test di unità
- **Test di sistema e di collaudo:** verifica che il sistema in cui andrà installato il software rispetti i requisiti richiesti, o che il software riesca ad adattarsi correttamente al contesto dell'azienda proponente. Il collaudo sarà sul software installato, finito il quale avverrà il rilascio del prodotto
- **Test di regressione:** nel caso di una modifica ad un singolo componente, andranno effettuati nuovamente tutti i test di unità e, se necessario, di integrazione riferiti a quel componente

2.5.3 Strategie per la verifica

Le strategie per effettuare operazioni di verifica prevedono quattro fasi principali che verranno di seguito descritte:

- **Verifica di accettabilità:** per l'accettazione, Team Committed si prefiggerà di testare che le funzionalità illustrate all'interno dei requisiti siano coperte dagli appositi test di unità ed integrazione oltre che da prove funzionali sul sistema.

- **Verifica del disegno architetturale:** Dopo aver realizzato lo schema architetturale del prodotto verrà verificato che siano rispettati alcuni precisi principi di seguito elencati:

- Layering: le classi devono rispettare una struttura gerarchica
- Packaging: le classi devono essere accorpate con una certa logica
- Coesione: una unità è progettata in modo tale da rispondere unicamente ai requisiti ad essa associati nel tracciamento dei requisiti
- Accoppiamento ridotto: le classi saranno progettate in modo tale che siano il più possibile indipendenti

Questa fase di verifica verrà eseguita anche grazie all’ausilio di apposite metriche descritte in “Misure e Metriche” (2.5.4)

- **Verifica del codice:** La verifica del codice ha diversi obiettivi, il primo può essere identificato nel trovare, segnalare ed isolare quelle parti di codice che non sono conformi alle norme stabilite in precedenza nel documento Norme di progetto, attraverso una fase di analisi statica. Un altro obiettivo può essere l’identificazione di quelle parti di codice soggette ad errori di programmazione mediante analisi dinamica.
- **Verifica di performance e criticità:** le performance e la criticità sono elementi che dipendono fortemente dall’infrastruttura di rete e dai server su cui risiederà l’applicazione. Team Committed si impegna quindi ad effettuare le dovute misurazioni riguardanti il carico di lavoro e la stabilità del sistema nel caso di picchi di richieste, questo per verificare la robustezza e reattività del sistema.

2.5.4 Misure e Metriche

Le misure e le metriche che adotteremo per la verifica della qualità del software si ispireranno alle indicazioni dello standard *ISO-14598*. Sono misure spesso focalizzate al miglioramento della capacità di prevedere e contenere il costo del software, Il *Team Committed* le utilizzerà per valutare la qualità del prodotto sia nel processo di progettazione che di codifica.

- **Complessità ciclomatica:** la complessità ciclomatica di un metodo è il numero di cammini linearmente indipendenti attraverso il codice sorgente. Per esempio, se il codice sorgente non contiene if o for, allora il livello di complessità sarà 1, poiché esiste un solo cammino
 - **Misure nella progettazione:** permettono di stimare il costo di un software e la sua qualità.
 - * **Numero di classi, coesione tra di esse e peso.** il peso di una classe è identificato dalla somma della complessità ciclomatica di tutti i metodi appartenenti alla classe

- * **Complessità di flusso:** misura la quantità di informazioni in entrata ed uscita da una funzione (fan in e fan out)
- **Misure sul codice**
 - * **Linee di codice**
 - * **Misure di coesione funzionale:** misurano le istanze di definizione e utilizzo di variabili e costanti (1994). Un numero elevato di parametri può essere abbassato, e quindi migliorato, raggruppando parametri tra loro correlati in classi diverse, aumentando manutenibilità e astrazione del codice
 - * **Livello di copertura di istruzioni, dei rami, dei percorsi base:** una non buona copertura del codice è indice di scarsa qualità dello stesso, inoltre vanno evitati eccessivi annidamenti dei metodi

3 Obbiettivi di qualità

Il *Team Committed* ha ritenuto importante fissare, di comune accordo, alcuni obiettivi di qualità da perseguire e raggiungere sia nei processi di realizzazione che del prodotto in se, questo per garantire una migliore e più efficace soddisfazione dei requisiti richiesti nel capitolato d'appalto.

3.1 Qualità dei processi

La realizzazione di prodotti di buona qualità richiede che vi sia qualità anche nei processi necessari. Il *Team Committed* ha quindi deciso di adottare lo standard **ISO/IEC 15504:1998 SPICE** (*Software Process Improvement and Capability Determination*) che definisce il modello denominato **SPA-I** (*Software Process Assessment & Improvement*) che offre metodi per la valutazione ed aumento di qualità dei processi.

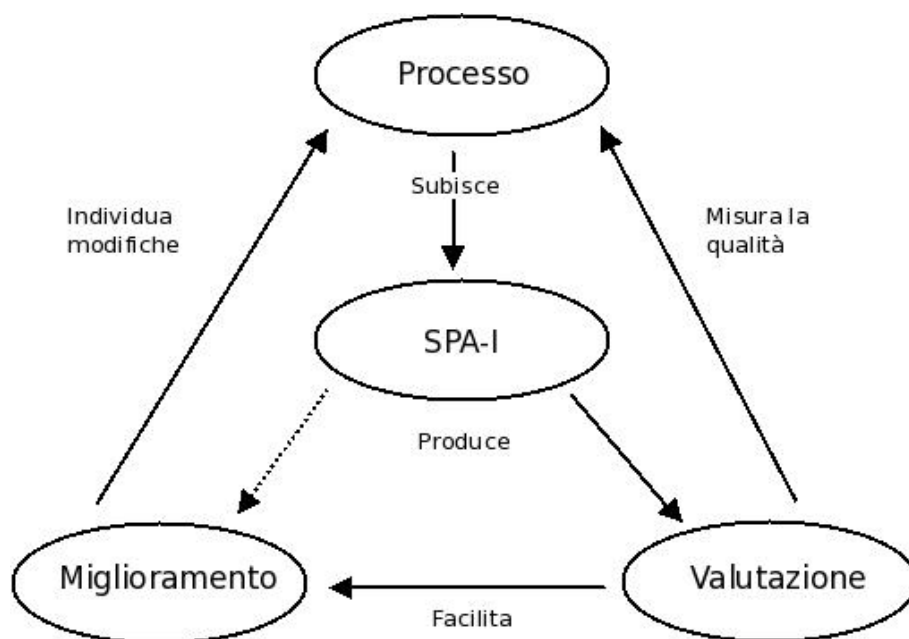


Figure 1: *Schema SPA-I*

Tutti i processi dovranno quindi essere sottoposti a valutazione in modo da verificarne la qualità ed eventualmente facilitarne il miglioramento. A tale scopo lo *SPICE* definisce nove attributi di processo per effettuare una migliore valutazione:

1. **Process performance:** un processo raggiunge i suoi obiettivi nel momento in cui trasforma dell'input identificabile in output identificabile

2. **Performance Management:** l'attuazione di un processo è pianificata e controllata al fine di generare risultati che rispondono agli obiettivi attesi
3. **Work Product Management:** l'attuazione di un processo è pianificata e controllata al fine di generare risultati che siano adeguatamente documentati, controllati e verificati
4. **Process Definition:** l'attuazione di un processo si basa su approcci standardizzati
5. **Process Resource:** il processo può contare sulle adeguate risorse umane, d'infrastrutture ecc. per essere attuato
6. **Process Measurement:** i risultati conseguiti e le misure rilevate durante l'attuazione di un processo sono utilizzati per assicurarsi che l'attuazione di tale processo supporti efficacemente il raggiungimento di obiettivi specifici
7. **Process Control:** un processo è controllato tramite la raccolta, analisi ed utilizzo delle misure di prodotto e di processo rilevate, con l'obiettivo di correggere, se necessario, le sue modalità di attuazione
8. **Process Change:** le modifiche alla definizione, gestione e attuazione di un processo sono controllate
9. **Continuous Integration:** le modifiche ad un processo sono identificate ed implementate con lo scopo di assicurare il continuo miglioramento nel raggiungere gli obiettivi rilevati per l'organizzazione

La norma stabilisce poi quattro differenti livelli di possesso di ciascuno degli attributi:

- **N - Non posseduto** (0 - 15% di possesso): non c'è evidenza oppure ce n'è poca del possesso di un attributo
- **P - Parzialmente posseduto** (16 - 50% di possesso): c'è evidenza di approccio sistematico al raggiungimento del possesso di un attributo e del raggiungimento di tale possesso, ma alcuni aspetti del possesso possono essere non prevedibili
- **L - Largamente posseduto** (51 - 85% di possesso): vi è evidenza di approccio sistematico al raggiungimento del possesso di un attributo e di un significativo livello di possesso di tale attributo, ma l'attuazione del processo può variare nelle diverse unità operative della organizzazione
- **F - (Fully) Pienamente posseduto** (86 - 100% di possesso): vi è evidenza di un totale e sistematico approccio e di un completo raggiungimento del possesso dell'attributo e non esistono significative differenze nel modo di attuare il processo tra le diverse unità operative

Vi sono poi vari livelli di maturità dei processi che sono dati dal diverso livello di possesso degli attributi:

- **Livello 0 - Processo incompleto:** il processo non è implementato o non raggiunge gli obiettivi. Non vi è evidenza di approcci sistematici agli attributi definiti
- **Livello 1 - Processo semplicemente attuato:** il processo viene messo in atto e raggiunge i suoi obiettivi. Non vi è evidenza di approcci sistematici agli attributi definiti. Il raggiungimento di questo livello è dimostrato attraverso il possesso degli attributi di “Process performance”
- **Livello 2 - Processo gestito:** il processo è attuato, ma anche pianificato, tracciato, verificato ed aggiustato se necessario, sulla base di obiettivi ben definiti. Il raggiungimento di questo livello è dimostrato attraverso il possesso degli attributi di “Performance management” e “Work product management”
- **Livello 3 - Processo definito:** il processo è attuato, pianificato e controllato sulla base di procedure ben definite, basate sui principi del software engineering. Il raggiungimento di questo livello è dimostrato attraverso il possesso degli attributi di “Process definition” e “Process resource”
- **Livello 4 - Processo predicibile:** il processo è stabilizzato ed è attuato all’interno di definiti limiti riguardo i risultati attesi, le performance, le risorse impiegate ecc. Il raggiungimento di questo livello è dimostrato attraverso il possesso degli attributi di “Process measurement” e “Process control”
- **Livello 5 - Processo ottimizzante:** il processo è predicibile ed in grado di adattarsi per raggiungere obiettivi specifici e rilevanti per l’organizzazione. Il raggiungimento di questo livello è dimostrato attraverso il possesso degli attributi di “Process change” e “Continuous integration”

L’applicazione dello standard *ISO/IEC 15504* porta a benefici sia agli sviluppatori del software che ai suoi utilizzatori o acquirenti. Per gli sviluppatori porta vantaggi nell’ottimizzazione dell’uso delle risorse, un contenimento dei costi, una maggiore tempestività di consegna del prodotto ultimato, migliore stima dei rischi e degli impegni e la possibilità di confrontarsi con delle *best practice*. Per gli utenti invece abbiamo una maggior facilità nella selezione dei fornitori, una migliore valutazione dei rischi di progetto, controllo dello stato di avanzamento in corso d’opera, riduzione dei costi di correzione degli errori ed un controllo dei rischi e delle varianti in corso d’opera.

3.2 Qualità del prodotto

Per quanto concerne la qualità del prodotto si è scelto di seguire alcune linee guida dettate dallo standard *ISO/IEC 9126*, classificato da sei caratteristiche generali e varie sotto-caratteristiche.

3.2.1 Funzionalità

Il prodotto software realizzato deve essere in grado di offrire apposite funzionalità che siano in grado di soddisfare le esigenze stabilite, necessarie per operare sotto condizioni specifiche.

- **Appropriatezza:** costituisce la capacità del software di offrire un insieme di funzioni per i compiti ed obiettivi prefissati all'utente
- **Accuratezza:** capacità del software di fornire i risultati concordati o i precisi effetti richiesti
- **Interoperabilità:** capacità del prodotto software di interagire ed operare con specifici sistemi
- **Conformità:** capacità del prodotto software di aderire a standard, convenzioni e regolamentazioni rilevanti al settore operativo a cui vengono applicate
- **Sicurezza:** la capacità del prodotto software di proteggere informazioni e dati impedendo che persone o sistemi non autorizzati possano accedervi o modificarli, mentre garantire queste operazioni a utenti o sistemi autorizzati

La misurazione del raggiungimento di questo obiettivo si calcolerà verificando la quantità di requisiti soddisfatti che avranno un riscontro in elementi funzionanti nell'applicazione prodotta. La soglia di sufficienza sarà quindi data dal soddisfacimento di tutti i requisiti obbligatori previsti dal capitolato d'appalto.

3.2.2 Affidabilità

L'affidabilità misura la capacità di un prodotto software di mantenere un determinato livello di prestazioni se usato in determinate condizioni e per un certo periodo.

- **Maturità:** capacità del prodotto software di evitare che si verifichino errori, malfunzionamenti o siano prodotti risultati non coerenti
- **Tolleranza agli errori:** capacità di mantenere determinati livelli di prestazioni nonostante la presenza di errori, malfunzionamenti o usi scorretti del prodotto
- **Recuperabilità:** capacità di un prodotto di ripristinare il livello appropriato di prestazioni e di essere in grado di recuperare le informazioni rilevanti in seguito ad un malfunzionamento
- **Aderenza:** capacità di aderire a standard, regole e convenzioni inerenti l'affidabilità

La misurazione del raggiungimento di questo obiettivo si calcolerà confrontando il numero di esecuzioni totale con quelle andate a buon fine e che hanno mantenuto un livello di prestazioni tali da poter permettere l'utilizzo previsto del prodotto.

3.2.3 Efficienza

La capacità di fornire adeguate prestazioni in relazione alla quantità di risorse usate determina l'efficienza.

- **Comportamento rispetto al tempo:** capacità di fornire adeguati tempi di risposta, elaborazione e velocità, sotto condizioni determinate
- **Utilizzo delle risorse:** capacità di utilizzare in maniera adeguata la giusta quantità e tipologia di risorse
- **Conformità:** capacità di aderire a standard e specifiche sull'efficienza

La misurazione del raggiungimento di questo obiettivo sarà determinata dal tempo necessario per ottenere una risposta dal servizio sia in condizioni normali che in condizioni di sovraccarico.

3.2.4 Usabilità

La capacità di un prodotto software di essere capito, appreso e usato dall'utente in certe condizioni determina la sua usabilità.

- **Comprensibilità:** costituisce la facilità di comprensione dei concetti del prodotto, permettendo all'utente quindi di comprendere se il software è appropriato
- **Apprendibilità:** capacità di diminuire l'impegno richiesto agli utenti per imparare ad utilizzare l'applicazione
- **Operabilità:** capacità di porre gli utenti in condizioni tali da utilizzare il prodotto per i propri scopi e controllarne l'uso
- **Attrattiva:** capacità del software di essere piacevole per l'utente
- **Conformità:** capacità del software di aderire a standard o convenzioni relativi all'usabilità

La misurazione del raggiungimento di questo obiettivo sarà costituita dalla capacità dell'applicativo di adattarsi ai vari tipi di ambienti in cui esso verrà eseguito come ambienti desktop o dispositivi mobile. L'usabilità sarà poi ritenuta raggiunta fornendo un'interfaccia il più possibile chiara, semplice ed intuitiva e rendendo l'applicativo fruibile a quelle categorie di utenti affetti da disabilità come daltonici o ipovedenti.

3.2.5 Manutenibilità

La manutenibilità rappresenta la capacità del software di essere modificato, includendo correzioni, miglioramenti o adattamenti.

- **Facilità di analisi:** rappresenta la facilità con la quale è possibile analizzare il codice per localizzare un eventuale errore
- **Modificabilità:** capacità del prodotto software di permettere l'implementazione di una specificata modifica
- **Stabilità:** capacità del software di evitare effetti inaspettati derivanti da modifiche errate
- **Testabilità:** capacità del software di essere facilmente testato per validare le modifiche apportate

La misurazione del raggiungimento di questo obiettivo saranno legate al rispetto delle misure metriche descritte nel capitolo 2.5.3

3.2.6 Portabilità

La portabilità è la capacità di un software d'essere portato da un ambiente di lavoro ad un altro.

- **Adattabilità:** capacità del software di essere adattato per differenti ambienti operativi senza dover applicare modifiche diverse da quelle fornite per il software considerato
- **Installabilità:** capacità del software di essere installato in uno specifico ambiente
- **Conformità:** capacità del software di aderire a standard e convenzioni relative alla portabilità
- **Sostituibilità:** capacità di essere utilizzato al posto di un altro software per svolgere gli stessi compiti

L'applicazione desktop deve essere installabile in tutti i principali sistemi operativi disponibili (Mac OSX, Windows, GNU Linux). Il front-end deve essere compatibile e funzionante con tutti i browser che aderiscono al W3C e con i browser di dispositivi mobile di diverso tipo (iOS, Windows Phone, Android, Symbian). Infine l'applicazione Android deve essere compatibile con il maggior numero di distribuzioni possibile del sistema operativo.

4 Gestione amministrativa della revisione

4.1 Comunicazione e risoluzione di anomalie

Un'anomalia consiste in un comportamento non coerente con le aspettative di qualità prefissate.

Lo strumento scelto per la gestione delle anomalie è la sezione “*Issue*” messa a disposizione da Github. Coerentemente con l'organizzazione generale delle strategie di verifica, nuove anomalie potranno essere scoperte in due modi:

1. Al rilascio di ogni nuova versione con relativo registro delle modifiche, il Verificatore effettuerà una fase di verifica cercando eventuali anomalie.
2. Grazie all'approccio “*Broken Window Theory*”, chiunque in qualunque momento è incentivato alla ricerca di possibili anomalie.

Appena un'anomalia viene trovata è opportuna una segnalazione tempestiva attraverso i ticket.

Struttura di un ticket

La struttura di un ticket dovrà seguire le seguenti specifiche:

- Nome
- Descrizione
- Label
- Milestone
- Assegnee

A riguardo è possibile leggere in dettaglio le specifiche di un ticket nelle **Norme di Progetto - V2.0**.

Risoluzione di un ticket Le modalità di risoluzione e chiusura di un ticket sono descritte nelle **Norme di Progetto - V2.0**, sono volte a garantire la massima capacità di coordinamento fra i vari membri del gruppo garantendo un'attenzione alla qualità generale.

4.2 Trattamento delle discrepanze

Una **discrepanza** è un tipo di anomalia non grave, non concerne il corretto funzionamento del prodotto, ma può riguardare un allontanamento dai requisiti attesi specificati nel capitolato d'appalto oppure una violazione delle *Norme di Progetto*.

Le modalità di ricerca e comunicazione delle discrepanze è del tutto simile alle modalità specificate per le anomalie; la risoluzione presenta invece modalità differenti.

Una volta creato il ticket sarà compito del verificatore riconoscere quale tipo di discrepanza si tratta e nel caso riguardi una violazione delle *Norme di Progetto* provvederà a comunicare il problema all'Amministratore che prenderà provvedimenti. Se invece il problema riguarda un allontanamento dai requisiti, una volta identificata l'origine della discrepanza, il Verificatore solleciterà l'Analista per valutare la gravità e i costi per risolverla.

4.3 Procedure di controllo di qualità di processo

Le procedure di gestione della qualità che il gruppo *Team Committed* userà si basano sul concetto di “**ciclo di Deming**” o **PDCA**, utili a garantire un continuo processo di miglioramento basandosi sulla comunicazione attiva da parte delle varie componenti del gruppo, una continua connessione fra le fasi di analisi, progettazione, produzione e verifica o collaudo. Lo scopo è un continuo miglioramento dei processi qualitativi unito ad una riduzione di costi e sprechi.

Il concetto è molto vicino al termine **Kaizen**, comunemente tradotto con “*miglioramento continuo*”, si riferisce ai processi di miglioramento qualitativo che devono coinvolgere tutto il gruppo di persone al lavoro. Si basa sulla filosofia de “*L'energia viene dal basso*”, ovvero sull'idea che il processo qualitativo non dipenda solo dall'organizzazione di un progetto, ma anche sul lavoro fatto sui singoli processi e prodotti.

5 Pianificazione ed esecuzione del collaudo

5.1 Test di Sistema

Tale sezione viene adibita alla pianificazione ed illustrazione dei test di sistema che verranno attuati prima del rilascio del prodotto ultimato.

Lo scopo dell'eseguire tali test è dimostrare che l'applicazione prodotta soddisfa le richieste del cliente, fornendo quindi esempi di test da attuare per confermare che i requisiti individuati nel documento **Analisi Dei Requisiti - V2.0** sono stati soddisfatti.

5.1.1 Ambito Generale

Cod. requisito	Cod. verifica	Modalità verifica	Stato verifica
RFOB 1	TS-RFOB 1	Prova dinamica: il server su cui risiede il sistema dovrà dimostrarsi funzionante rispondendo alle richieste dei vari componenti	Pianificata
RFOB 1.1	TS-RFOB 1.1	Prova dinamica: viene verificato testando la possibilità di connettersi al servizio da differenti dispositivi	Pianificata
RFD 2	TS-RFD 2	Prova dinamica: viene verificato se, al raggiungimento di determinati obiettivi di <u>Gamification</u> , il sistema assegnerà effettivamente dei badge/trofei	Pianificata
RPOB 1	TS-RPOB 1	Prova dinamica: viene verificato confrontando i tempi di risposta del servizio in caso di basso traffico della rete con i tempi di risposta in caso di traffico elevato. I due tempi di risposta dovranno quindi essere simili	Pianificata
RPOB 2	TS-RPOB 2	Prova dinamica: viene verificato confrontando i tempi di risposta del servizio tramite l'utilizzo di un database test di domande risiedente nel medesimo server con quelli di un database test di domande risiedente in un server remoto. I due tempi di risposta dovranno essere simili	Pianificata
RQOP 3.1	TS-RQOP 3.1	Prova dinamica: viene verificato se, al passaggio del mouse sopra determinati campi o sezioni del servizio, questo sia in grado di restituire appositi consigli sull'utilizzo dello stesso	Pianificata

RQOP 3.2	TS-RQOP 3.2	Prova dinamica: viene verificato se, da ogni schermata, il sistema renderà possibile accedere ad un'apposita area di aiuto	Pianificata
RVOP 1.1	TS-RVOP 1.1	Prova dinamica: viene verificato testando l'applicativo desktop su sistemi operativi di diverso tipo	Pianificata

5.1.2 Ambito Utente

Cod. requisito	Cod. verifica	Modalità verifica	Stato verifica
RFOB 3	TS-RFOB 3	Prova dinamica: viene verificato se il sistema effettivamente registra la data e l'ora dell'esecuzione delle varie azioni previste all'interno di un file di log	Pianificata
RFOB 4	TS-RFOB 4	Prova dinamica: viene verificato se il sistema offrirà effettivamente la possibilità all'Utente che ha smarrito la password di connettersi all'apposita pagina per richiederne il recupero	Pianificata
RFOB 4.1	TS-RFOB 4.1	Prova dinamica: viene verificato che, se la procedura di richiesta modifica password è stata eseguita correttamente, il sistema invierà effettivamente una mail contenente una password temporanea generata randomicamente	Pianificata
RFOB 4.1.1	TS-RFOB 4.1.1	Prova dinamica: viene verificato che al primo login dopo il recupero password, il sistema chieda effettivamente che questa venga modificata	Pianificata
RFOB 11	TS-RFOB 11	Prova dinamica: viene verificato che il sistema invii una mail di conferma qual'ora i dati personali dell'Utente siano stati modificati	Pianificata
RFOB 11.1	TS-RFOB 11.1	Prova dinamica: viene verificato che il sistema invii una mail di conferma qual'ora venisse modificata la password di accesso	Pianificata

5.1.3 Ambito Dipendente

Cod. requisito	Cod. verifica	Modalità verifica	Stato verifica
RFOB 5	TS-RFOB 5a	Prova dinamica: viene verificato se a fronte dell'inserimento di un indirizzo email o password non validi il sistema risponda in maniera negativa	Pianificata
RFOB 5	TS-RFOB 5b	Prova dinamica: viene verificato se a fronte dell'inserimento di un indirizzo email e password validi il sistema risponda in maniera positiva	Pianificata
RFOB 5.1	TS-RFOB 5.1	Prova dinamica: viene verificato se il sistema permetterà effettivamente di effettuare il login tramite l'apposita applicazione web, desktop o android	Pianificata
RFOB 6	TS-RFOB 6	Prova dinamica: viene verificato che il sistema sia in grado di fornire, grazie ad un timer, periodiche domande precariate al <u>Dipendente</u>	Pianificata
RFOB 7	TS-RFOB 7	Prova dinamica: viene verificato che il sistema sia in grado di fornire delle domande al <u>Dipendente</u> qual'ora le richiedesse	Pianificata
RFOB 7.1	TS-RFOB 7.1	Prova dinamica: viene verificato che il sistema limiti, grazie ad un timer, il numero di domande che il <u>Dipendente</u> può richiedere	Pianificata
RFOB 10	TS-RFOB 10a	Prova dinamica: viene verificato che il sistema sia in grado di offrire la possibilità al <u>Dipendente</u> di accedere all'area di modifica dei propri dati personali	Pianificata
RFOB 10	TS-RFOB 10b	Prova dinamica: viene verificato che il sistema risponda negativamente alla modifica delle informazioni qual'ora fossero stati inseriti dati non corretti	Pianificata
RFOB 10	TS-RFOB 10c	Prova dinamica: viene verificato che il sistema risponda positivamente alla modifica delle informazioni qual'ora fossero stati inseriti dati corretti	Pianificata
RFOB 11.2	TS-RFOB 11.2	Prova dinamica: viene verificato che il sistema invii una mail di conferma qual'ora venisse modificato l'indirizzo email	Pianificata

RFOB 12	TS-RFOB 12	Prova dinamica: viene verificato se un <u>Dipendente</u> autenticato è in grado di accedere ai propri dati personali	Pianificata
RFOB 13	TS-RFOB 13a	Prova dinamica: viene verificato se il sistema risponderà positivamente alla scelta di un <u>Utente</u> di rispondere ad una domanda proposta	Pianificata
RFOB 13	TS-RFOB 13b	Prova dinamica: viene verificato se il sistema risponderà positivamente alla scelta di un <u>Utente</u> di rinviare una domanda proposta	Pianificata
RFOB 14.1	TS-RFOB 14.1	Prova dinamica: viene verificato se il sistema è in grado di riconoscere ed offrire informazioni sulla tipologia di domanda (nuova, posticipata, sbagliata)	Pianificata
RFOB 15	TS-RFOB 15	Prova dinamica: viene verificato se un <u>Dipendente</u> sarà in grado di visualizzare lo storico dei propri punteggi	Pianificata
RFOB 16	TS-RFOB 16	Prova dinamica: viene verificato se un <u>Dipendente</u> sarà in grado di poter terminare la propria sessione	Pianificata

5.1.4 Ambito Amministratore Azienda

Cod. requisito	Cod. verifica	Modalità verifica	Stato verifica
RFOB 17	TS-RFOB 17a	Prova dinamica: viene verificato se il sistema risponde positivamente all'inserimento di username e password validi nella sezione dedicata al login di un <u>Amministratore Azienda</u>	Pianificata
RFOB 17	TS-RFOB 17b	Prova dinamica: viene verificato se il sistema risponde negativamente all'inserimento di username e password non validi nella sezione dedicata al login di un <u>Amministratore Azienda</u>	Pianificata
RFOB 18	TS-RFOB 18	Prova dinamica: viene verificato se il sistema è in grado di offrire all' <u>Amministratore Azienda Autenticato</u> la possibilità di visionare la pagina di gestione dipendenti	Pianificata

RFOB 19	TS-RFOB 19a	Prova dinamica: viene verificato se il sistema risponderà positivamente all'inserimento, da parte di un <u>Amministratore Azienda Autenticato</u> , di dati corretti relativi ad un nuovo <u>Dipendente</u>	Pianificata
RFOB 19	TS-RFOB 19b	Prova dinamica: viene verificato se il sistema risponderà negativamente all'inserimento, da parte di un <u>Amministratore Azienda Autenticato</u> , di dati non corretti relativi ad un nuovo <u>Dipendente</u>	Pianificata
RFOB 19.2	TS-RFOB 19.2	Prova dinamica: viene verificato se il sistema, dopo l'inserimento di un nuovo <u>Dipendente</u> da parte dell' <u>Amministratore Azienda Autenticato</u> , invierà una mail contenente i dati del nuovo account all'indirizzo specificato	Pianificata
RFOB 19.2.1	TS-RFOB 19.2.1	Prova dinamica: viene verificato se il sistema inserirà nella mail inviata l'username del nuovo account <u>Dipendente</u>	Pianificata
RFOB 19.2.2	TS-RFOB 19.2.2	Prova dinamica: viene verificato se il sistema inserirà nella mail inviata la password del nuovo account <u>Dipendente</u>	Pianificata
RFOB 20	TS-RFOB 20	Prova dinamica: viene verificato se il sistema sarà in grado di far eliminare all' <u>Amministratore Azienda Autenticato</u> un account <u>Dipendente</u>	Pianificata
RFOB 21	TS-RFOB 21a	Prova dinamica: viene verificato che il sistema sia in grado di permettere all' <u>Amministratore Azienda Autenticato</u> di accedere alla pagina di modifica degli account dei Dipendenti	Pianificata
RFOB 21	TS-RFOB 21b	Prova dinamica: viene verificato che il sistema risponda positivamente alla modifica dei dati di un <u>Dipendente</u> da parte dell' <u>Amministratore Azienda Autenticato</u> con informazioni valide	Pianificata

RFOB 21	TS-RFOB 21c	Prova dinamica: viene verificato che il sistema risponda negativamente alla modifica dei dati di un <u>Dipendente</u> da parte dell' <u>Amministratore Azienda Autenticato</u> con informazioni non valide	Pianificata
RFOB 22	TS-RFOB 22	Prova dinamica: viene verificato che il sistema permetta ad un <u>Amministratore Azienda Autenticato</u> di visionare le informazioni di ogni <u>Dipendente</u> registrato	Pianificata
RFOB 23	TS-RFOB 23	Prova dinamica: viene verificato che il sistema permetta da un <u>Amministratore Azienda Autenticato</u> di visionare le informazioni relative ai punteggi di ogni <u>Dipendente</u> registrato	Pianificata
RFOB 24	TS-RFOB 24	Prova dinamica: viene verificato che il sistema permetta all' <u>Amministratore Azienda Autenticato</u> di modificare i badge/trofei di ogni singolo <u>Dipendente</u>	Pianificata
RFOB 25	TS-RFOB 25	Prova dinamica: viene verificato che il sistema permetta all' <u>Amministratore Azienda Autenticato</u> di terminare la propria sessione	Pianificata

5.1.5 Ambito Amministratore Sicurezza

Cod. requisito	Cod. verifica	Modalità verifica	Stato verifica
RFOB 26	TS-RFOB 26a	Prova dinamica: viene verificato se il sistema risponde positivamente all'inserimento di username e password validi nella sezione dedicata al login di un <u>Amministratore Sicurezza</u>	Pianificata
RFOB 26	TS-RFOB 26b	Prova dinamica: viene verificato se il sistema risponde negativamente all'inserimento di username e password non validi nella sezione dedicata al login di un <u>Amministratore Sicurezza</u>	Pianificata

RFOB 27	TS-RFOB 27	Prova dinamica: viene verificato che il sistema offra la possibilità ad un <u>Amministratore Sicurezza Autenticato</u> di consultare le domande dell'azienda	Pianificata
RFOB 28	TS-RFOB 28	Prova dinamica: viene verificato che il sistema offra la possibilità ad un <u>Amministratore Sicurezza Autenticato</u> di accedere all'interfaccia per la modifica delle domande di un'azienda	Pianificata
RFOB 28.1	TS-RFOB 28.1	Prova dinamica: viene verificato che il sistema offra la possibilità all' <u>Amministratore Sicurezza Autenticato</u> di aggiungere nuove domande da sottoporre ai dipendenti da un'apposito database generale	Pianificata
RFOB 28.2	TS-RFOB 28.2	Prova dinamica: viene verificato che il sistema offra la possibilità all' <u>Amministratore Sicurezza Autenticato</u> di eliminare domande dalla lista di quelle che possono essere somministrate ai <u>Dipendenti</u>	Pianificata
RFOP 29	TS-RFOP 29	Prova dinamica: viene verificato che il sistema offra la possibilità all' <u>Amministratore Sicurezza Autenticato</u> di poter modificare gli obiettivi di <u>Gamification</u>	Pianificata
RFOP 29.1	TS-RFOP 29.1	Prova dinamica: viene verificato che il sistema esegua correttamente la modifica dei badge/trofei assegnabili ai dipendenti al raggiungimento di obiettivi di <u>Gamification</u>	Pianificata
RFOB 30	TS-RFOB 30	Prova dinamica: viene verificato che il sistema esegua permetta all' <u>Amministratore Sicurezza Autenticato</u> di terminare correttamente la propria sessione	Pianificata

5.1.6 Ambito Amministratore Installatore

Cod. requisito	Cod. verifica	Modalità verifica	Stato verifica
----------------	---------------	-------------------	----------------

RFOB 31	TS-RFOB 31	Prova dinamica: viene verificato che sia possibile eseguire una corretta installazione del sistema in alcuni dispositivi fissi di test	Pianificata
RFOB 32	TS-RFOB 32	Prova dinamica: viene verificato che sia possibile eseguire una corretta installazione del sistema in alcuni dispositivi mobili di test	Pianificata
RFOB 33	TS-RFOB 33	Prova dinamica: viene verificato che sia possibile una corretta installazione del servizio web in un server test	Pianificata
RFOB 34	TS-RFOB 34	Prova dinamica: viene verificato che sia possibile una corretta creazione di un database test per garantire il corretto funzionamento del sistema	Pianificata
RFOP 35	TS-RFOP 35a	Prova dinamica: viene verificato che il sistema risponda positivamente all'inserimento di una domanda con dati corretti	Pianificata
RFOP 35	TS-RFOP 35b	Prova dinamica: viene verificato che il sistema negativamente all'inserimento di una domanda con dati non corretti	Pianificata
RFOP 35	TS-RFOP 35c	Prova dinamica: viene verificato che il sistema risponda positivamente alla modifica di una domanda con l'inserimento di dati corretti	Pianificata
RFOP 35	TS-RFOP 35d	Prova dinamica: viene verificato che il sistema risponda negativamente alla modifica di una domanda con l'inserimento di dati non corretti	Pianificata
RFOP 35	TS-RFOP 35e	Prova dinamica: viene verificato che il sistema permetta all' <u>Amministratore Installatore</u> di eliminare correttamente una domanda	Pianificata

5.2 Test di integrazione

In questa sezione verrà descritta la pianificazione dei test di verifica e collaudo che verranno eseguiti per le varie parti del sistema. Grazie a questi test si vuole verificare il corretto funzionamento del software, i test sono divisi fra test intra-componenti e inter-componenti.

5.2.1 Test intra-componenti

I componenti verranno testati e collaudati seguendo una strategia incrementale attraverso il principio di bottom-up.

Verranno prima collaudati i componenti con minori dipendenze funzionali, per poi proseguire con l'albero delle dipendenze. Nella fattispecie prima sarà necessario avere funzionanti tutti i database e la gestione degli stessi, poi si potrà proseguire con la restante parte back-end e la parte front-end.

Per ogni componente verranno effettuate prove dalle quali ci si può aspettare un esito positivo o un esito negativo.

Componente	Funzionalità da verificare
CDESK1 (login <u>Dipendente</u>)	<ul style="list-style-type: none">• Login tramite username e password• Recupero password dimenticata
CDESK2 (notifica nuova domanda)	<ul style="list-style-type: none">• Notifica nuova domanda secondo i tempi impostati• Rifiuto o accettazione nuova domanda• Apertura browser se viene accettata la domanda
CWEB1 (login)	<ul style="list-style-type: none">• Login tramite username e password• Riconoscimento tipologia <u>Utente</u>• Recupero password dimenticata
CWEB2 (gestione dati <u>Utente</u>)	<ul style="list-style-type: none">• Visualizzazione dati <u>Utente</u> loggato• Modifica degli stessi
CWEB3 (amministrazione utenti)	<ul style="list-style-type: none">• Inserimento nuovi utenti• Modifica utenti• Eliminazione utenti

CWEB4 (rispondere ad una domanda)	<ul style="list-style-type: none"> • Visualizzazione della domanda • Gestione risposta, aggiornamento punteggi • Visualizzazione esito risposta
CWEB5 (amministrazione domande)	<ul style="list-style-type: none"> • Inserimento e rimozione delle domande relative ad un'azienda dall'insieme delle domande globali
CWEB6 (visualizzazione punteggi)	<ul style="list-style-type: none"> • Visualizzazione propri punteggi aggiornati
CWEB7 (gestione punteggi)	<ul style="list-style-type: none"> • Visualizzazione punteggi globali
CWEB8 (notifica nuova domanda)	<ul style="list-style-type: none"> • Notifica nuova domanda • Rifiuto o accettazione nuova domanda
CMOB1 (login)	<ul style="list-style-type: none"> • Login tramite username e password • Recupero password dimenticata
CMOB2 (gestione dati <u>Utente</u>)	<ul style="list-style-type: none"> • Visualizzazione dati <u>Utente</u> loggato • Modifica degli stessi
CMOB3 (rispondere nuova domanda)	<ul style="list-style-type: none"> • Visualizzazione della domanda • Gestione risposta, aggiornamento punteggi • Visualizzazione esito risposta

CMOB4 (visualizzazione punteggi)	<ul style="list-style-type: none"> • Visualizzazione punteggi aggiornati
CMOB5 (notifica nuova domanda)	<ul style="list-style-type: none"> • Visualizzazione nuova domanda
CBACK1 (login)	<ul style="list-style-type: none"> • Login tramite username e password • Recupero password dimenticata
CBACK2 (amministrazione utenti)	<ul style="list-style-type: none"> • Visualizzazione dati utenti • Modifica degli stessi
CBACK3 (gestione domande e risposte)	<ul style="list-style-type: none"> • Inserimento nuove domande • Modifica domande • Eliminazione domande
CBACK4 (amministrazione punteggi)	<ul style="list-style-type: none"> • Visualizzazione punteggi globali • Inserimento e modifica criteri di attribuzione punteggi e badge

5.2.2 Test inter-componenti

Man mano che il collaudo prosegue verranno aggiunti test e prove sui componenti in maniera incrementale. Ad ogni nuovo collaudo quindi verrà testato il corretto funzionamento inter-componenti tenedono conto delle loro dipendenze. Il sistema software in ogni caso non prevede grosse dipendenze fra i vari componenti front-end e back-end, il grosso del lavoro inter-componenti relativo alle dipendenze andrà rivolto al corretto funzionamento del database e delle relative richieste.

6 Dettaglio dell'esito delle revisioni

Ad ogni revisione che il gruppo deve sostenere, dovrà essere presentata al committente l'apposita documentazione prodotta corredata da una presentazione orale di quanto realizzato allo scopo di valutare lo stato di avanzamento dei lavori. Il committente a questo punto produrrà una valutazione dettagliata del lavoro svolto, documento per documento. Gli eventuali errori ed inesattezze che sorgono da tale valutazione, così come le migliorie proposte, dovranno produrre una correzione e modifica dei documenti in modo tale da evitare il perpetuarsi di inesattezze nelle revisioni successive.

6.1 Revisione dei Requisiti

Il lavoro svolto per la fase di Revisione dei Requisiti è stato valutato positivamente, dimostrando che il materiale prodotto era sufficientemente adeguato. Sono state comunque segnalati alcuni errori e problematiche principalmente nei documenti di Analisi dei Requisiti, Piano di Progetto, Piano di Qualifica e Norme di Progetto; di seguito verranno quindi riportate nel dettaglio le operazioni di correzione:

- **Analisi dei requisiti:** Sono stati completamente rivisti e modificati i diagrammi dei casi d'uso e le relative descrizioni, eliminando gli errori e le imprecisioni riscontrate. Sono stati poi di conseguenza eliminati, modificati o aggiunti alcuni requisiti ed è stato rieffettuato infine il tracciamento Casi d'uso - Requisiti.
- **Piano di Progetto:** Ridefinito l'inizio delle attività di verifica, ora correttamente distribuite lungo tutto l'arco del progetto. Unificate in un'unica sezione la Pianificazione ed il Preventivo economico.
- **Piano di Qualifica:** Approfondite le operazioni di inspection nei documenti e nel codice. Migliorata la descrizione dell'organizzazione delle strategie di verifica ed aggiunto uno specifico capitolo inerente alle strategie di verifica vere e proprie. Migliorata la descrizione della fase di verifica svolta nella fase di Revisione dei Requisiti.
- **Norme di Progetto:** Corretta la notazione della data. Dettagliato maggiormente il documento per quanto riguarda le procedure e gli strumenti utilizzati.

7 Resoconto attività di verifica

7.1 Revisione dei Requisiti

Nella fase di Revisione dei Requisiti è stata effettuata un'attività di verifica sui documenti prodotti. Nel dettaglio è stato effettuata un'analisi statica come indicato nella sezione 2.5.2 tramite walkthrough prima ed inspection poi, verificando che i documenti rispettino i punti individuati e riportati in tale sezione inerenti la formattazione del testo, effettuando le dovute procedure per la correzione degli errori rilevati. Nel dettaglio, i documenti sono stati revisionato effettuando quindi il controllo ortografico tramite Aspell, il plugin per LyX, mentre il controllo grammaticale, sintattico e lessicale è avvenuto tramite un'accurata rilettura da parte dei revisori. Sono stati poi controllati anche i contenuti tabellari e grafici. La segnalazione di irregolarità è avvenuta tramite Ticket che sono state prese in carico successivamente dal redattore e risolte.

Infine, per effettuare il doppio tracciamento, si sono controllate le apposite tabelle come descritto nel capitolo 2.1.

7.2 Revisione di Progettazione

Anche in tale fase è stata effettuata un'accurata analisi statica sui documenti in maniera simile a quanto avvenuto nella fase di Revisione dei Requisiti. Una particolare attenzione è stata però posta nella fase di verifica dei diagrammi UML in quanto la gran parte degli errori e delle imprecisioni rilevate nei documenti presentati durante la precedente revisione erano concentrati in tale sezione. Nei documenti di Specifica Tecnica e Piano di Qualifica è poi stato fondamentale effettuare un'adeguata ed attenta fase di tracciamento. Come in precedenza sono stati infine utilizzati i Ticket per segnalare eventuali irregolarità ed errori che sono stati poi presi in carico dal redattore e risolti.