



Piano di Qualifica

Gruppo SteakHolders — Progetto MaaP

Informazioni sul documento

Versione	3.0.0
Redazione	Enrico Rotundo Giacomo Fornari Nicolò Tresoldi
Verifica	Gianluca Donato
Approvazione	Luca De Franceschi
Uso	Esterno
Distribuzione	Prof. Tullio Vardanega Prof. Riccardo Cardin Gruppo SteakHolders CoffeeStrap

Descrizione

Questo documento descrive le operazioni di verifica e validazione seguite dal gruppo SteakHolders relativi al progetto MaaP.



Registro delle modifiche

Versione	Data	Persone coinvolte	Descrizione
3.0.0	2014-02-05	Luca De Franceschi (Responsabile)	Approvazione.
2.2.0	2014-01-03	Gianluca Donato (Verificatore)	Verifica
2.1.3	2014-01-29	Nicolò Tresoldi (Amministratore)	Stesura dell'appendice "PDCA"
2.1.2	2014-01-26	Giacomo Fornari (Amministratore)	Aggiunte metriche architettura software
2.1.1	2014-01-20	Serena Girardi (Verificatore)	Correzioni indicate nella verifica
2.1.0	2014-01-17	Luca De Franceschi (Verificatore)	Verifica
2.0.2	2014-01-15	Enrico Rotundo (Verificatore)	Correzioni in base a tabella di valutazione interna al gruppo.
2.0.1	2014-01-14	Enrico Rotundo (Amministratore)	Correzioni struttura documento.
2.0.0	2014-01-10	Nicolò Tresoldi (Amministratore)	Aggiornamento sistema di versionamento.
1.3.1	2013-12-20	Nicolò Tresoldi (Responsabile in deroga)	Approvazione.
1.2.3	2013-12-19	Giacomo Fornari (Verificatore)	Verifica.
1.2.2	2013-12-19	Gianluca Donato (Verificatore)	Stesura resoconto verifiche.
1.2.1	2013-12-17	Giacomo Fornari (Verificatore)	Verifica.
1.1.7	2013-12-15	Enrico Rotundo (Progettista)	Stesura Pianificazione strategica.
1.1.6	2013-12-14	Enrico Rotundo (Progettista)	Stesura Gestione amministrativa.
1.1.5	2013-12-13	Enrico Rotundo (Progettista)	Stesura Appendice Qualità.
1.1.4	2013-12-12	Enrico Rotundo (Progettista)	Stesura Misure e Metriche.
1.1.3	2013-12-11	Enrico Rotundo (Progettista)	Stesura Tecniche.
1.1.2	2013-12-10	Enrico Rotundo (Progettista)	Stesura Introduzione.
1.1.1	2013-12-10	Enrico Rotundo (Progettista)	Creazione documento.



Indice

1	Introduzione	4
1.1	Scopo del documento	4
1.2	Scopo del prodotto	4
1.3	Glossario	4
1.4	Riferimenti	5
1.4.1	Normativi	5
1.4.2	Informativi	5
2	Visione generale della strategia di verifica	6
2.1	Definizione obiettivi	6
2.1.1	Qualità di processo	6
2.1.2	Qualità di prodotto	6
2.2	Procedure di controllo di qualità di processo	6
2.3	Procedure di controllo di qualità di prodotto	7
2.4	Organizzazione	7
2.5	Strategia	7
2.6	Responsabilità	8
2.7	Risorse	8
2.8	Misure e Metriche	8
2.8.1	Metriche per i processi	8
2.8.1.1	Schedule Variance	8
2.8.1.2	Budget Variance	9
2.8.1.3	Produttività	9
2.8.1.4	Impegno	10
2.8.1.5	Modifiche	10
2.8.1.6	Copertura dei test	10
2.8.2	Metriche per i documenti	11
2.8.2.1	Gulpease	11
2.8.3	Metriche per il software	11
2.8.3.1	Complessità ciclomatica	11
2.8.3.2	Densità della complessità ciclomatica	12
2.8.3.3	Numero di metodi - NOM	12
2.8.3.4	Bug_G per lines of code	12
2.8.3.5	Variabili non utilizzate e non definite	13
2.8.3.6	Numero parametri per metodo	13
2.8.3.7	Numero funzioni d'interfaccia per package	13
2.8.3.8	$Halstead_G$	13
2.8.3.9	Maintainability index	15
2.8.3.10	First-order density	15
2.8.3.11	Change cost	15
3	Gestione amministrativa della revisione	16
3.1	Comunicazione delle anomalie	16
3.2	Controlli per la qualità di processo	16
4	Pianificazione ed esecuzione del collaudo	18



Appendici	19
A Pianificazione dei test	19
A.1 Livelli di testing	19
A.2 Test di sistema	20
A.3 Test di integrazione	28
A.4 Test di validazione	30
B Resoconto delle attività di verifica	37
B.1 Riassunto delle attività di verifica	37
B.1.1 Revisione dei Requisiti	37
B.1.1.1 Miglioramenti	37
B.1.2 Revisione dei Requisiti	37
B.2 Dettaglio delle verifiche tramite analisi	38
B.2.1 Analisi	38
B.2.1.1 Documenti	38
B.2.2 Progettazione Architetturale	38
B.2.2.1 Documenti	38
B.3 Dettaglio dell'esito delle revisioni	39
B.3.1 Revisione dei Requisiti	39
C Qualità	40
C.1 Qualità dei processi	41
C.2 Qualità del prodotto software	42
D PDCA	43
D.1 Revisione dei requisiti	43
D.2 Revisione di progettazione	43

Elenco delle tabelle

A.1 Tracciamento Test di Sistema - Requisiti	27
A.2 Descrizione test d'Integrazione	30
A.3 Tracciamento Test di Validazione - Requisiti	36
A.4 Problemi pre RR individuati e relative soluzioni.	38
A.5 Esiti verifica documenti, Analisi	38
A.6 Esiti verifica documenti, Progettazione Architetturale	39
A.7 Risultati metriche per i processi, Revisione di progettazione	43
A.8 Risultati metriche per i processi, Revisione di progettazione	44

Elenco delle figure

1 Il ciclo di miglioramento dei processi	17
2 V-Model per il testing software	20
3 Sequenza d'integrazione delle componenti	28
4 Diagramma di attività dei test	29
5 Continuous quality improvement with PDCA	40



1 Introduzione

L'obiettivo primario è la *qualità_G* del prodotto e dei suoi processi, ottenibile tramite una serie di controlli proattivi stabiliti al tempo zero. L'assenza di tali verifiche abbinata ad un team di più soggetti senza particolari accortezze e competenze, portano al progressivo deterioramento del materiale prodotto, sia esso codice sorgente o documentazione. Questo fenomeno è noto sotto il nome di *Broken windows theory_G* ed è intrinseco alla componente sociale dell'uomo. Il concetto chiave è prevenire l'inserimento di materiale non aderente alle *Norme di Progetto v3.0.0* poiché, secondo la teoria succitata, innescherebbe un meccanismo che deteriora la qualità all'interno del *repository_G*.

Si vogliono gestire le componenti *accidentali_G* dei processi, ossia tutte quelle problematiche non intrinseche alla produzione, ma che ne sono direttamente collegate; si desidera scongiurare il pericolo di operare *by correction_G* per evitare modifiche in corso d'opera che possono bloccare la maturazione del prodotto e richiedere dispendiose correzioni.

1.1 Scopo del documento

Il Piano di Qualifica illustra la strategia di verifica e validazione che il gruppo SteakHolders ha deciso di adottare. È necessario dare una dimensione alla qualità dei prodotti e dei processi, operazioni che non rientrano nei normali ruoli di progetto, bensì rappresentano una *funzione aziendale*. Vi sono molteplici punti di vista della qualità, il *committente_G* sarà in grado di valutare su basi oggettive quanto prodotto. Inoltre, la direzione del progetto potrà fare affidamento sulla consistenza dello stato dello stesso e il proponente avrà una solida base di verifica ideata e funzionante.

Questo documento si colloca nella parte relativa al *Project* delle *quattro P del Software Engineering* (People, Product, Project, Process) perché tratta una parte fondamentale sulle attività a supporto della produzione di MaaP.

1.2 Scopo del prodotto

Lo scopo del progetto è la realizzazione di un *framework_G* per generare interfacce web di amministrazione dei dati di *business_G* basato su *stack_G* *Node.js_G* e *MongoDB_G*. L'obiettivo è quello di semplificare il processo di implementazione di tali interfacce che lo sviluppatore, appoggiandosi alla produttività del framework MaaP, potrà generare in maniera semplice e veloce ottenendo quindi un considerevole risparmio di tempo e di sforzo. Il fruitore finale delle pagine generate sarà infine l'esperto di business che potrà visualizzare, gestire e modificare le varie entità e dati residenti in *MongoDB_G*. Il prodotto atteso si chiama *MaaP_G* ossia *MongoDB as an admin Platform*.

1.3 Glossario

Ogni occorrenza di termini tecnici, di dominio e gli acronimi sono marcati con una G in pedice e riportati nel documento *Glossario v3.0.0*.



1.4 Riferimenti

Vengono elencati i riferimenti su cui si è basata l'organizzazione dell'attività di qualifica.

1.4.1 Normativi

- **Norme di Progetto:** *Norme di Progetto*;
- **Capitolato d'appalto C1:** <http://www.math.unipd.it/~tullio/IS-1/2013/Progetto/C1.pdf>.

1.4.2 Informativi

- **Piano di Progetto:** *Piano di Progetto v3.0.0*;
- **Slide di Ingegneria del Software mod. A:** <http://www.math.unipd.it/~tullio/IS-1/2013/>;
- **SWEBOK 2004 Version - capitolo 5 e 11:** <http://www.computer.org/portal/web/swebok/htmlformat>;
- **Software Engineering 9th - I. Sommerville (Pearson, 2011) - capitoli: 8, 24, 26:** <http://www.pearsoned.co.uk/bookshop/detail.asp?item=100000000377819>;
- **ISO/IEC 15504** http://en.wikipedia.org/wiki/ISO/IEC_15504;
- **ISO/IEC 9126** http://en.wikipedia.org/wiki/ISO/IEC_9126.



2 Visione generale della strategia di verifica

La strategia generale adottata è quella di automatizzare il più possibile il lavoro di verifica; questo richiede scelta e uso di $tools_G$ adeguatamente configurati. L'obiettivo è avere un riscontro affidabile e numericamente trattabile che permetta di assicurare il grado di qualità predeterminato. Il lavoro manuale verrà così ridotto al minimo e confinato all'opera di validazione. La speranza è che dei buoni processi portino ad un buon software.

2.1 Definizione obiettivi

2.1.1 Qualità di processo

La qualità del processo è un fattore determinante per la qualità del prodotto. Si è deciso di perseguire la qualità servendosi di due modelli:

- $SPICE_G$ ¹, definito nello standard ISO/IEC 15504, ai fini di una valutazione oggettiva dei processi, per darne un giudizio di maturità e per individuare azioni migliorative;
- $PDCA_G$ ², per il controllo delle attività di processo ripetibili e misurabili e per la manutenibilità dei processi stessi incrementandone la qualità.

2.1.2 Qualità di prodotto

Oltre alla qualità di processo, sono necessari degli obiettivi rivolti direttamente alla qualità del prodotto per massimizzare l'efficacia. A tal fine, lo standard ISO/IEC 9126³ classifica la qualità del software e definisce delle metriche per la sua misurazione.

2.2 Procedure di controllo di qualità di processo

Le linee guida per la gestione della qualità di processo seguono il modello $PDCA_G$ descrivendo come devono essere attuate le procedure di controllo:

- La pianificazione deve essere dettagliata;
- Le attività pianificate devono essere monitorate;
- Le risorse necessarie per conseguire gli obiettivi devono essere definite;
- Il controllo deve servirsi delle metriche per verificare il miglioramento della qualità del processo.

La pianificazione delle attività volte al miglioramento continuo dei processi sono descritte nel *Piano di Progetto v3.0.0*.

¹Si veda appendice C.1 per approfondimenti

²Si veda appendice C per approfondimenti

³Si veda appendice C.2 per approfondimenti

2.3 Procedure di controllo di qualità di prodotto

Il controllo per la qualità di prodotto definisce i seguenti processi:

- **Software Quality Assurance** (SQA_G): deve assicurare che i processi pianificati siano appropriati e successivamente implementati secondo la pianificazione, e che siano forniti sistemi di misurazione dei processi. Tale processo deve essere preventivo invece che correttivo;
- **Verifica**: si occupa di accertare che l'esecuzione delle attività di processi svolti nel periodo in esame non abbia introdotto errori nel prodotto. La verifica viene svolta sui prodotti dei processi per accertare il rispetto delle regole, delle convenzioni e delle procedure;
- **Validazione**: si occupa di accertare che i prodotti finali realizzati siano conformi alle attese.

2.4 Organizzazione

Viene verificata la qualità di ogni processo e di ogni output da essi prodotti. Ogni periodo descritto nel *Piano di Progetto v3.0.0* produce output di diverso tipo, per questo è necessario programmare le attività di verifica in modo mirato:

- **Analisi**: in questo periodo si controlla che i processi e la documentazione prodotta rispettino le *Norme di Progetto v3.0.0* e verrà verificato che ogni requisito abbia corrispondenza in un caso d'uso;
- **Progettazione Architettuale**: in questo periodo si verificano i processi incrementali relativi all'analisi e ai nuovi documenti di progettazione, e che i test siano adeguatamente pianificati come descritto nel *Piano di Progetto v3.0.0* ed eseguiti secondo quanto descritto nelle *Norme di Progetto v3.0.0*;
- **Progettazione di Dettaglio e Codifica**: in questo periodo vanno verificati i processi incrementali relativi alla progettazione assieme alla verifica delle attività di codifica grazie a tecniche di analisi statica e dinamica.

Il *Diario delle modifiche* viene incluso in ogni documento al fine di tracciarne uno storico dell'evoluzione.

2.5 Strategia

Il *Piano di Progetto* fissa una serie di scadenze improrogabili, pertanto è necessario definire con chiarezza una strategia di qualifica efficace. Gli incrementi sulla documentazione o sul codice possono essere di natura programmata, quindi prefissati nel calendario, oppure possono insorgere come inaspettati. In questo caso sarà necessario programmare le dovute modifiche; è questo il caso di *bug_G* o *errori* (vedi paragrafo 3.1). La qualità di ogni incremento si basa sul fatto che la struttura di qualifica garantisce il rispetto delle *Norme di Progetto v3.0.0*. Questo lavoro verrà svolto con l'aiuto di automatismi che segnaleranno le problematiche rilevate in modo da permettere una rapida correzione. L'utilizzo di software apposito permette di eseguire controlli mirati senza consumare risorse umane. L'implementazione di tali controlli viene descritta nelle *Norme di Progetto v3.0.0*.



2.6 Responsabilità

La responsabilità della verifica viene attribuita al *Responsabile* di progetto e ai *Verificatori*. I compiti e le modalità di attuazione sono definiti nel *Piano di Progetto v3.0.0*.

2.7 Risorse

La qualifica dei processi, essendo un processo, consuma delle risorse, che si dividono in due categorie:

- **Umane:** le figure coinvolte sono il *Responsabile* di progetto e il *Verificatore*. I processi da loro effettuati consumano ore di produttività contabilizzate e schedate secondo il *Piano di Progetto v3.0.0*. Le ore di produttività sono fissate dalle regole di progetto (<http://www.math.unipd.it/~tullio/IS-1/2013/Progetto/PD01b.html>) in un minimo di 85 e un massimo di 105 ore individuali. Il *Piano di Progetto v3.0.0* determina la distribuzione di tali quote orarie con la relativa retribuzione. Ai fini della qualifica si potrà parlare di ore di produttività tralasciandone l'aspetto economico, in quanto non rientra nel dominio del documento succitato;
- **Tecnologiche:** riguardano i *mezzi* utilizzati per gli automatismi per la qualità e la loro gestione. Trattandosi esclusivamente di mezzi informatici, vengono consumate unità di calcolo considerate a costo nullo. Tale considerazione si basa sul fatto che tutti i tipi di elaborazioni informatiche sono svolte su mezzi per i quali non è richiesto né un contributo economico, né un quantitativo temporale abbastanza consistente da poter essere considerato degno di nota.

Le modalità del loro impiego sono descritte dettagliatamente nelle *Norme di Progetto v3.0.0*.

2.8 Misure e Metriche

Vengono adottate delle metriche per rendere misurabili e valutabili i processi, i documenti ed il software prodotto. La visione non vuole essere comparativa, ma serve al gruppo per monitorare l'andamento dei processi e la qualità del prodotto.

2.8.1 Metriche per i processi

Le seguenti metriche rappresentano un indicatore volto a monitorare e prevedere l'andamento delle principali variabili critiche del progetto (i tempi e i costi). Sono state scelte metriche di tipo *consuntivo_G* perché danno un riscontro immediato sullo stato attuale del progetto; ad ogni incremento verranno valutati tali indici e, se necessario, verranno stabiliti opportuni provvedimenti da parte del *Responsabile* di progetto.

2.8.1.1 Schedule Variance



Indica se si è in linea, in anticipo o in ritardo rispetto alla schedulazione delle attività di progetto pianificate.

$$SV = BCWP - BCWS$$

Dove *BCWP* indica il valore delle attività realizzate alla data corrente e *BCWS* rappresenta il costo pianificato per realizzare le attività di progetto alla data corrente. È un indicatore di efficacia soprattutto nei confronti del Cliente. Se $SV > 0$ significa che il progetto sta procedendo con maggior velocità a quanto pianificato, viceversa se negativo.

2.8.1.2 Budget Variance

Indica se alla data corrente si è speso di più o di meno rispetto a quanto previsto.

$$BV = BCWS - ACWP$$

Dove *BCWS* indica il costo pianificato per realizzare le attività di progetto alla data corrente e *ACWP* rappresenta il costo effettivamente sostenuto alla data corrente. È un indicatore che ha un valore unicamente contabile e finanziario. Se $BV > 0$ significa che il progetto sta spendendo il proprio budget con minor velocità di quanto pianificato, viceversa se negativo.

2.8.1.3 Produttività

Produttività di documentazione Indica la produttività media di documentazione delle risorse impiegate, cioè delle persone coinvolte, nei diversi stadi del progetto.

$$\text{Produttività di documentazione} = \text{Parole} / \text{Ore persona}$$

Dove *Parole* indica il numero di parole presenti nei documenti e *Ore persona* rappresenta il numero di ore produttive dei componenti del gruppo.

Parametri utilizzati:

- Range-ottimale: $[\geq 100]$.

Produttività di test Indica la produttività media dei test realizzati.

$$\text{Produttività di test} = \text{Numero di test} / \text{Ore persona}$$

Dove *Numeroditest* indica il numero di test eseguiti e *Ore persona* rappresenta il numero di ore produttive dei componenti del gruppo.

I parametri utilizzati per questa metrica verranno stabiliti alla entro l'inizio dell'esecuzione dei test.



Produttività di codifica Indica la produttività media delle attività di codifica.

$$\text{Produttività di codifica} = \text{LOCs} / \text{Ore persona}$$

Dove *LOCs* indica il numero di linee di codice prodotte e *Ore persona* rappresenta il numero di ore produttive dei componenti del gruppo.

I parametri utilizzati per questa metrica verranno stabiliti alla entro l'inizio del processo di codifica.

2.8.1.4 Impegno

Indica l'impegno richiesto dal gruppo per la realizzazione del progetto.

$$\text{Impegno} = \text{Dimensione} / \text{Produttività}$$

Dove *Dimensione* indica il tempo produttivo impiegato e *Produttività* rappresenta la media delle produttività totali (di documentazione, di test, di codifica).

Parametri utilizzati:

- Range-ottimale: $[\geq 0,6]$.

2.8.1.5 Modifiche

Indica quante modifiche sono state approvate dal responsabile. Le modifiche possono riguardare requisiti, funzionalità, codice e documenti.

$$\text{Modifiche} = \text{Numero di modifiche}$$

Dove *Numero di modifiche* indica le issue etichettate come "richiesta di modifica" e "approvate".

Parametri utilizzati:

- Range-accettazione: $[0 - 20]$;
- Range-ottimale: $[0 - 10]$.

2.8.1.6 Copertura dei test

Indica la percentuale di casi coperti da test eseguiti.

$$\text{Copertura del test} = \text{Numero di funzioni testate} * 100 / \text{Numero totale di funzioni disponibili}$$

Parametri utilizzati:

- Range-accettazione: $[70 - 100]$;
- Range-ottimale: $[80 - 100]$.

2.8.2 Metriche per i documenti

La *leggibilità* dei documenti è indispensabile per garantirne la qualità. Si è scelto di adottare un indice per misurare la leggibilità dei testi in lingua italiana:

2.8.2.1 Gulpease

L'Indice Gulpease è un indice di leggibilità di un testo tarato sulla lingua italiana. Rispetto ad altri ha il vantaggio di utilizzare la lunghezza delle parole in lettere anziché in sillabe, semplificandone il calcolo automatico. Permette di misurare la complessità dello stile di un documento. L'indice di Gulpease considera due variabili linguistiche: la lunghezza della parola e la lunghezza della frase rispetto al numero delle lettere. La formula per il suo calcolo è:

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

I risultati sono compresi tra 0 e 100, dove il valore 100 indica la leggibilità più alta e 0 la leggibilità più bassa. In generale risulta che testi con un indice:

- Inferiore a 80 sono difficili da leggere per chi ha la licenza elementare;
- Inferiore a 60 sono difficili da leggere per chi ha la licenza media;
- Inferiore a 40 sono difficili da leggere per chi ha un diploma superiore.

Parametri utilizzati:

- Range-accettazione: [40 - 100];
- Range-ottimale: [50 - 100].

2.8.3 Metriche per il software

La prima release di *Node.js_G* risale a Maggio 2009. È stata riscontrata una forte differenza tra le metriche disponibili per l'analisi statica rispetto a quelle per i linguaggi meno recenti. Inoltre nessun membro del gruppo ha conoscenza di tale linguaggio e delle sue particolarità, come l'aspetto *funzionale_G*. Tali differenze con i linguaggi studiati nel percorso universitario si sono tradotte nella difficoltà di individuare metriche non incentrate sulla visione ad oggetti del codice. Infine si è osservata l'assenza di strumenti per la misurazione di metriche tradizionali come la coesione e l'instabilità dei package. Di seguito vengono elencate le metriche per il software prodotto.

2.8.3.1 Complessità ciclomatica

La complessità ciclomatica è una metrica software che indica la complessità di un programma misurando il numero di cammini linearmente indipendenti attraverso il grafo di controllo di flusso. Nel grafo sopracitato i *nodi* corrispondono a gruppi indivisibili di istruzioni, mentre gli *archi* connettono due nodi se il secondo gruppo di istruzioni può essere eseguito immediatamente dopo il primo gruppo. Questo indice può essere applicato indistintamente a singole funzioni,



moduli, metodi o package di un programma. Si vuole utilizzare tale metrica per limitare la complessità durante la fase di sviluppo. Durante il testing è utile per determinare il numero di casi di test necessari, infatti l'indice di complessità è un limite superiore al numero di test necessari per raggiungere il coverage completo del modulo testato. Inoltre, uno studio ha mostrato forti corrispondenze tra le metriche di complessità e il livello di coesione nei package presi in esame⁴.

Parametri utilizzati:

- Range-accettazione: [0 - 15];
- Range-ottimale: [0 - 10]⁵.

2.8.3.2 Densità della complessità ciclomatica

Proposta come modifica alla complessità ciclomatica, questa metrica semplicemente la riesprime come percentuale sulle linee di codice logiche. Un valore basso è migliore di un valore alto.

2.8.3.3 Numero di metodi - NOM

Il *Number of methods* è una metrica usata per calcolare la media delle occorrenze dei metodi per package. Un package non dovrebbe contenere un numero eccessivo di metodi. Valori superiori al range ottimale massimo potrebbero indicare una necessità di maggiore decomposizione del package.

Parametri utilizzati:

- Range-accettazione: [3 - 10];
- Range-ottimale: [3 - 7].

2.8.3.4 Bug_G per lines of code

Questa metrica misura il numero di bug_G trovati su un certo quantitativo di linee di codice. L'aumentare del sorgente implica un incremento delle probabilità di nascondere errori, per questo è bene mantenere il codice più chiaro e semplice possibile. Con la crescita del prodotto è utile monitorare il rapporto tra i difetti trovati e il codice incrementale. Tale indice dovrebbe restare costante o diminuire nel tempo. Il gruppo fissa questa metrica ad un massimo di 60, considerando il fatto che nessun membro ha conoscenze dello *stack tecnologico_G* utilizzato. L'obiettivo è di giungere alla *Revisione di Accettazione* con valori compresi tra 0 e 20. Lo sfioramento di tali valori determina l'intervento del *Responsabile* di progetto, che dovrà individuare tempestivamente la causa del problema.

⁴Stein, C., G. Cox and L. Etzkorn, 2005. Exploring the Relationship between Cohesion and Complexity. J. Comput. Sci., 1: 137-144.

⁵McCabe (dicembre 1976). A Complexity Measure. IEEE Transactions on Software Engineering: 308-320.

2.8.3.5 Variabili non utilizzate e non definite

La presenza di variabili non utilizzate viene considerata $pollution_G$ pertanto non viene tollerata. Tali occorrenze vengono rilevate analizzando l'*Abstract syntax tree_G* (AST) eseguendo una cross-reference tra le variabili dichiarate e quelle inizializzate. Per sua natura, *Javascript_G* non blocca l'insorgenza di tali occorrenze, pertanto si rischia di dichiarare una variabile e poi utilizzarne una con nome leggermente diverso, oppure semplicemente dichiarare una variabile che in seguito non verrà mai utilizzata.

Parametri utilizzati:

- Range-accettazione: [0 - 0];
- Range-ottimale: [0 - 0].

2.8.3.6 Numero parametri per metodo

Un numero elevato di parametri per un metodo potrebbe evidenziare un metodo troppo complesso.

Non c'è una regola forte per il numero di parametri possibili in un metodo o costruttore, citando Robert Martin, in Clean Code⁶ :

“The ideal number of arguments for a function is zero (niladic). Next comes one (monadic), followed closely by two (dyadic). Three arguments (triadic) should be avoided where possible. More than three (polyadic) requires very special justification – and then shouldn't be used anyway.”

e Steve McConnell, in Code Complete⁷

“limit the number of a routine's parameters to about seven, seven is a magic number for people's comprehension”

ci atteniamo come linea guida ai parametri sotto evidenziati.

Parametri utilizzati:

- Range-accettazione: [0 - 8];
- Range-ottimale: [0 - 4].

2.8.3.7 Numero funzioni d'interfaccia per package

Un numero elevato di funzioni d'interfaccia in un package evidenzia un possibile errore di progettazione.

Parametri utilizzati:

- Range-accettazione: [0 - 20];
- Range-ottimale: [1 - 10].

2.8.3.8 $Halstead_G$

⁶Robert Martin, Clean Code: A Handbook of Agile Software Craftsmanship. Prentice Hall (2008)

⁷Steve McConnell, Code Complete: A Practical Handbook of Software Construction. Microsoft Press (2004)



La metrica di $Halstead_G$ non è solamente un indice di complessità, ma identifica le proprietà misurabili del software e le relative relazioni. Si basa sull'osservazione che una metrica dovrebbe valutare l'implementazione di un algoritmo in linguaggi differenti ed essere indipendente dall'esecuzione su una specifica piattaforma.

In un problema vengono identificati:

- n_1 = il numero di operatori distinti
- n_2 = il numero di operandi distinti
- N_1 = il numero totale di operatori
- N_2 = il numero totale di operandi

Da cui vengono calcolati:

- $n = n_1 + n_2$: vocabolario della funzione
- $N = N_1 + N_2$: lunghezza della funzione

Data la bassa disponibilità nella rete di valori di riferimento, i range specificati sono frutto di un confronto tra il $report_G$ sulla complessità di una libreria *open source*_G presa come esempio (<https://github.com/philbooth/complexity-report/blob/master/EXAMPLE.md>) e i valori dichiarati in <http://www.mccabe.com/pdf/McCabeIQMetrics.pdf>. Tali valori vengono dichiarati momentanei (RR) e saranno da rivalutare sia considerando altre fonti, sia considerando i valori rilevati in parti del codice che il gruppo considera come riferimento.

$Halstead_G$ difficulty per-function Il livello di difficoltà di una funzione misura la propensione all'errore ed è proporzionale al numero di operatori presenti.

$$D = \left(\frac{n_1}{2}\right) * \left(\frac{N_2}{n_2}\right)$$

Parametri utilizzati:

- Range-accettazione: [0 - 30];
- Range-ottimale: [0 - 15].

$Halstead_G$ volume per-function Il volume descrive la dimensione dell'implementazione di un algoritmo e si basa sul numero di operazioni eseguite e sugli operandi di una funzione. Il volume di una function senza parametri composta da una sola linea è 20, mentre un indice superiore a 1000 indica che probabilmente la funzione esegue troppe operazioni.

$$V = N * \log_2 n$$

Parametri utilizzati:

- Range-accettazione: [20 - 1500];
- Range-ottimale: [20 - 1000].



$Halstead_G$ effort per-function Lo sforzo per implementare o comprendere il significato di una funzione è proporzionale al volume a al suo livello di difficoltà.

$$E = V * D$$

Parametri utilizzati:

- Range-accettazione: [0 - 400];
- Range-ottimale: [0 - 300].

2.8.3.9 Maintainability index

Questa metrica⁸ è una scala logaritmica da $-\infty$ a 171, calcolata sulla base delle linee di codice logiche, della complessità ciclomatica e dall'indice $Halstead_G$ effort. Un valore alto indica una maggiore manutenibilità.

2.8.3.10 First-order density

La percentuale di tutte le possibili dipendenze interne che si sono effettivamente realizzata nel progetto. Un valore basso è migliore di un valore alto.

2.8.3.11 Change cost

La percentuale media dei moduli condizionati dal cambiamento di un modulo nel progetto. Un valore basso è migliore di un valore alto.

⁸Definita nel 1991 da Paul Oman e Jack Hagemeister alla University of Idaho.

3 Gestione amministrativa della revisione

3.1 Comunicazione delle anomalie

Il processo di *Software Quality Management*_G è finalizzato alla ricerca dei difetti. L'identificazione delle anomalie ne permette la correzione e informa il *Responsabile* di progetto sullo stato del prodotto. Distinguere e catalogare le anomalie è utile per discutere durante revisioni e riunioni su che correzioni attuare e con quale priorità. Di seguito vengono elencate le definizioni di anomalie (IEEE 610.12-90) adottate dal gruppo:

- **Error**: differenza riscontrata tra il risultato di una computazione e il valore teorico atteso (e.g. uscita dal range di accettazione degli indici di misurazione);
- **Fault**: un passo, un processo o un dato definito in modo erraneo (e.g. violazioni di norme tipografiche da parte di un documento). Corrisponde a quanto viene definito come *bug*_G;
- **Failure**: il risultato di un *fault* (e.g. incongruenza del prodotto con funzionalità indicate nell'analisi dei requisiti, incongruenza del codice con il design del prodotto);
- **Mistake**: azione umana che produce un risultato errato (e.g. anomalie nel repository).

La catalogazione delle anomalie permette l'impostazione di metriche in grado di valutarne l'andamento e in alcuni casi di predirlo. In particolare è stata scelta la metrica che conteggia il numero di *bug*_G per *lines of code*. Il *SCR*_G (software change request) utilizzato dal gruppo viene individuato nelle *Norme di Progetto v3.0.0*.

3.2 Controlli per la qualità di processo

Le procedure di controllo per la qualità di processo sono finalizzate a migliorare la qualità del prodotto e/o diminuire i costi e tempi di sviluppo. Esistono due approcci principali:

- **A maturità di processo**: riflette le buone pratiche di management e tecniche di sviluppo. L'obiettivo primario è la qualità del prodotto e la prevedibilità dei processi;
- **Agile**: sviluppo iterativo senza l'overhead della documentazione e di tutti gli aspetti pre-determinabili. Ha come caratteristica la responsività ai cambiamenti dei requisiti cliente e uno sviluppo rapido.

Verrà utilizzato il primo approccio, in quanto più adatto ad un team inesperto. Con una visione proattiva si cerca di avere maggior controllo e previsione sulle attività da svolgere. Questa viene anche indicata come *best practice*_G per gruppi poco esperti.

Il processo con maggiore influenza sulla qualità del sistema non è quello di sviluppo ma quello di progettazione. È qui che le capacità e le esperienze dei singoli danno un contributo decisivo.

Il miglioramento dei processi è un processo ciclico composto da tre sotto-processi:

- Misurazione del processo: misura gli attributi del progetto, punta ad allineare gli obiettivi con le misurazioni effettuate. Questo forma una *baseline*_G che aiuta a capire se i miglioramenti hanno avuto effetto;
- Analisi del processo: vengono identificate le problematiche ed i colli di bottiglia dei processi;
- Modifiche del processo: i cambiamenti vengono proposti in risposta alle problematiche riscontrate.

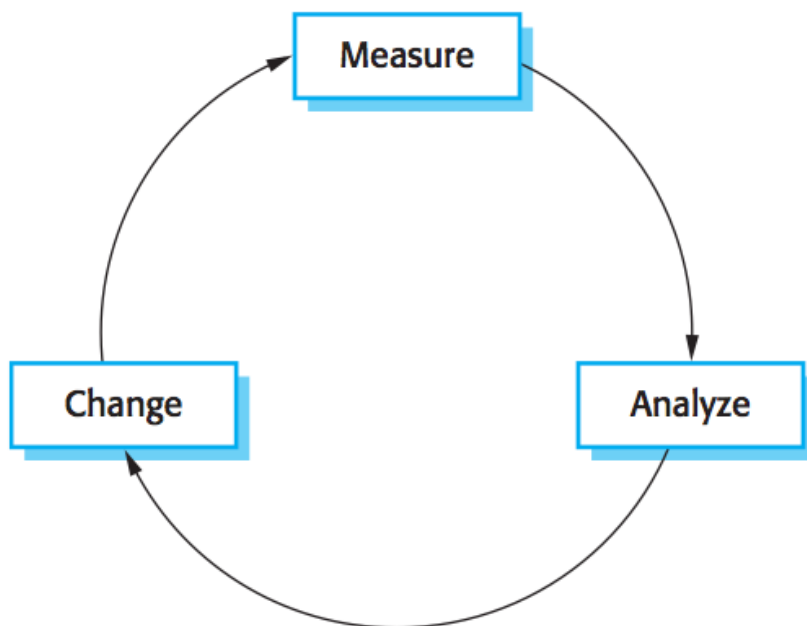


Figura 1: Il ciclo di miglioramento dei processi

Il gruppo procederà nel seguente modo:

- Nella sezione *Dettaglio delle verifiche tramite analisi* (B.2) verranno inserite le misurazioni rilevate sulle le metriche descritte in *Misure e Metriche* (2.8);
- L'analisi viene effettuata i giorni precedenti alle consegne previste dal committente; il *Riassunto delle attività di verifica* (B.1) contiene l'analisi del processo, le relative considerazioni comprendenti le problematiche riscontrate;
- Le modifiche al processo vengono attuate all'inizio del processo incrementale successivo. Queste attività sono programmate nel *Piano di Progetto v3.0.0*.



4 Pianificazione ed esecuzione del collaudo

Allo stato attuale non è possibile definire in dettaglio i collaudi in quanto non è stata affrontata la progettazione del prodotto, pertanto la pianificazione ed esecuzione dei collaudi saranno trattate nella prossima revisione.

A Pianificazione dei test

Si vuole adottare una strategia di verifica del software tramite test opportunamente predeterminati e che garantiscano almeno un test per ogni requisito. I test sono l'applicazione delle tecniche di verifica dinamica introdotte nelle *Norme di Progetto v3.0.0*; tali attività, oltre a richiedere l'esecuzione del programma, devono poter essere ripetibili, ossia tramite delle specifiche su come riprodurre i test vogliamo che il loro output sia deterministico. È importante che i test di unità vengano svolti in parallelo, dando precedenza alle unità che producono risultati utili alla comprensione del loro funzionamento integrato, l'ambiente di testing deve soddisfare tale obiettivo. L'attività di test deve produrre un \log_G che specifica quando e chi ha eseguito il test e con quali input; l'insorgenza di $failure_G$ deve essere tracciata e catalogata.

A.1 Livelli di testing

Il testing del software viene suddiviso in livelli differenti e si concretizzano in un'esecuzione bottom-up che avanza sequenzialmente alle attività di codifica e di validazione. I test che si andranno ad applicare sono di cinque tipi, riservando la specifica delle ultime due tipologie alla prossima revisione:

1. Test di Validazione (TV): viene verificato che il prodotto soddisfi quanto richiesto dal *proponente_G* individuando delle macro azioni da eseguire sul sistema che un normale utente svolge comunemente;
2. Test di Sistema (TS): sono test relativi al comportamento dell'intero sistema ossia viene verificato che la sua architettura generale funziona complessivamente bene;
3. Test di Integrazione (TI): vengono verificate le componenti del sistema contenute nella *Specifica Tecnica v3.0.0*, ossia viene verificato che i *package_G* siano funzionanti e in grado di funzionare nel loro insieme;
4. Test di Unità (TU): viene testata ogni unità, ossia la più piccola parte di lavoro assegnabile ad un programmatore. In questo progetto una unità dovrebbe corrispondere ad una `function` o a un `method`;
5. Test di Regression (TR): possono essere test di tutte le tipologie succitate che devono mostrare il funzionamento del prodotto a seguito di una modifica.

La figura 2 illustra come i test elencati vengono distribuiti durante in ciclo di vita del prodotto.

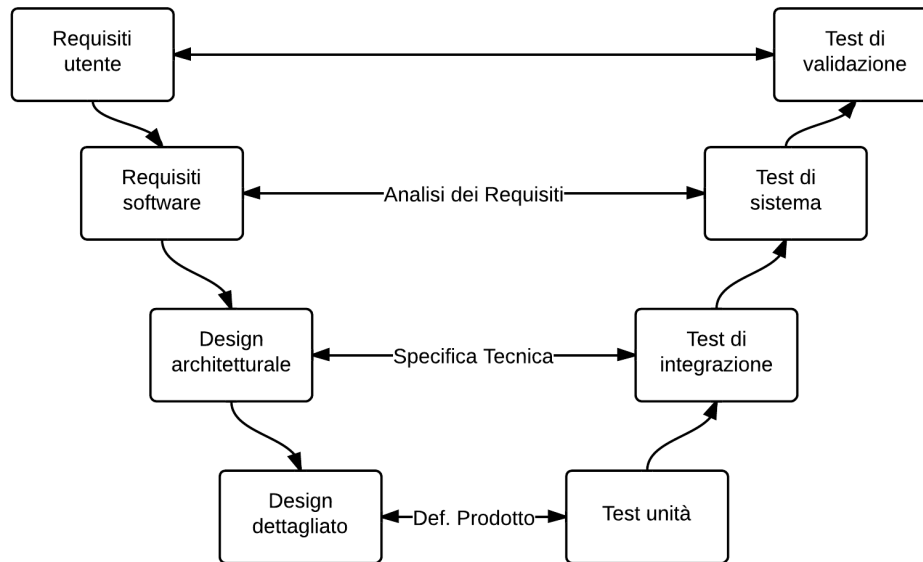


Figura 2: V-Model per il testing software

A.2 Test di sistema

Vengono qui descritti i test di sistema che andranno a verificare il funzionamento complessivo delle componenti. Nella seguente tabella, lo stato di ogni test è definito da N.E per non eseguito.



Test Sistema	Descrizione	Stato	Requisito
TS-RA1O 1.1	Verificare che durante la verifica delle credenziali l'indirizzo email venga immesso tramite un campo di testo apposito.	N.E	RA1O 1.1
TS-RA1O 1.2	Verificare che durante la verifica delle credenziali, la password venga immessa tramite un capo di testo apposito.	N.E	RA1O 1.2
TS-RA1O 1.3	Verificare che il sistema verifichi le credenziali di un utente tramite un database indipendente da quello che contiene la Collection.	N.E	RA1O 1.3
TS-RA1O 1.3.1	Verificare che, in caso di fallimento dell'autenticazione di un utente, il sistema visualizzi una pagina di errore.	N.E	RA1O 1.3.1
TS-RA1O 1.3.2	Verificare che in caso in cui autenticazione vada a buon fine, l'utente venga reindirizzato automaticamente sulla dashboard dell'applicazione.	N.E	RA1O 1.3.2
TS-RA1O 2.1	Verificare che il sistema permetta il recupero password attraverso l'inserimento dell'email.	N.E	RA1O 2.1
TS-RA1O 2.2	Verificare che un utente non autenticato che richiede il reset della propria password riceva un email con un link per il reset.	N.E	RA1O 2.2
TS-RA1O 2.3	Verificare che un utente non autenticato possa resettare la propria password tramite l'inserimento di una nuova password.	N.E	RA1O 2.3
TS-RA1O 4.1	Verificare che la visualizzazione di una Collection-index consista in una tabella le cui righe corrispondono ai document presenti nel database e le cui colonne siano i relativi attributi.	N.E	RA1O 4.1
TS-RA1O 4.1.1	Verificare che ogni riga della tabella corrispondente ad un Document abbia una chiave selezionabile che rimanda alla corrispondente pagina show.	N.E	RA1O 4.1.1
TS-RA1D 4.1.2	Verificare che l'admin possa eliminare un documento tramite un link rapido.	N.E	RA1D 4.1.2



TS-RA1D 4.1.3	Verificare che l'admin possa modificare un document della collection-index.	N.E	RA1D 4.1.3
TS-RA1D 4.2	Verificare che sia possibile visualizzare un sottoinsieme di Document tramite dei filtri personalizzati sugli attributi.	N.E	RA1D 4.2
TS-RA1F 4.3	Verificare che l'amministratore possa creare un nuovo Document nella base di dati.	N.E	RA1F 4.3
TS-RA1F 4.4	Verificare che sia possibile eseguire un'azione personalizzata tramite l'esecuzione di un pulsante.	N.E	RA1F 4.4
TS-RA1O 5.1	Verificare che l'admin possa editare ogni singolo attributo modificabile del documento della pagina show.	N.E	RA1O 5.1
TS-RA1F 5.2	Verificare che l'utente possa eseguire un'azione personalizzata tramite l'esecuzione di un pulsante.	N.E	RA1F 5.2
TS-RA1O 5.3	Viene verificato che l'utente possa eliminare il Document selezionato nella show-page.	N.E	RA1O 5.3
TS-RA1O 6.1	Viene verificato che l'admin possa creare un nuovo utente dalla pagina di amministrazione.	N.E	RA1O 6.1
TS-RA1O 6.1.1	Viene verificato che l'admin disponga di una pagina di creazione di un nuovo utente.	N.E	RA1O 6.1.1
TS-RA1O 6.1.1.1	Verificare che l'admin possa inserire l'indirizzo email del nuovo utente in un apposito campo di testo presente all'interno della pagina di creazione di un nuovo utente.	N.E	RA1O 6.1.1.1
TS-RA1O 6.1.1.2	Viene verificato che l'admin possa inserire la password del nuovo utente in un apposito campo di testo presente all'interno della pagina di creazione di un nuovo utente.	N.E	RA1O 6.1.1.2
TS-RA1O 6.1.1.3	Verificare che l'admin possa inserire il "livello utente" del nuovo utente tramite una combo-box presente all'interno della pagina di creazione di un nuovo utente.	N.E	RA1O 6.1.1.3



TS-RA1O 6.1.2	Viene verificato che l'applicazione prelevi tutti i dati inseriti dall'admin nella pagina di creazione di un nuovo utente e li invii al database delle credenziali, il quale provvederà all'inserimento del nuovo record.	N.E	RA1O 6.1.2
TS-RA1O 6.1.3	Verificare che venga visualizzato un messaggio d'errore nel caso in cui l'admin non abbia compilato correttamente i campi presenti all'interno della pagina di creazione di un nuovo utente.	N.E	RA1O 6.1.3
TS-RA1O 6.2	Viene verificato che l'admin abbia la possibilità di selezionare un utente dalla index-page e visualizzare la sua relativa show-page.	N.E	RA1O 6.2
TS-RA1O 6.2.1	Verificare che l'admin possa elevare l'utente normale selezionato al livello "admin" dalla show-page relativa.	N.E	RA1O 6.2.1
TS-RA1O 6.2.2	Verificare che l'admin possa declassare l'admin selezionato a livello di utente normale dalla show-page relativa.	N.E	RA1O 6.2.2
TS-RA1O 6.2.3	Viene verificato che l'admin possa modificare l'attributo email dell'utente selezionato dalla relativa show-page.	N.E	RA1O 6.2.3
TS-RA1O 6.2.4	Verificare che l'admin possa modificare l'attributo password dell'utente selezionato dalla relativa show-page.	N.E	RA1O 6.2.4
TS-RA1O 6.2.5	Viene verificato che l'admin possa eliminare l'utente visualizzato nella <i>show-page_G</i> .	N.E	RA1O 6.2.5
TS-RF1O 7	Verificare che il linguaggio DSL all'interno di MaaP Framework sia stato implementato e sia funzionante.	N.E	RF1O 7
TS-RF1O 8.1	Verificare che Maap Framework generi automaticamente lo scheletro dell'applicazione creata dallo sviluppatore.	N.E	RF1O 8.1



TS-RF1O 8.1.1	Verificare che Maap Framework importi automaticamente in un'apposita directory del progetto tutte le librerie necessarie al corretto funzionamento del sistema. Librerie necessarie: <ul style="list-style-type: none">• Express v-3.4.8• MongoDB v-1.3.23• Mongoose v-3.8.4	N.E	RF1O 8.1.1
TS-RF1O 8.1.2	Verificare che Maap Framework crei automaticamente in un'apposita directory il file di configurazione di default dell'applicazione generata.	N.E	RF1O 8.1.2
TS-RF1O 8.1.3	Viene verificato che Maap Framework crei automaticamente il sistema di autenticazione per l'applicazione generata.	N.E	RF1O 8.1.3
TS-RF1O 8.1.4	Verificare che Maap Framework crei automaticamente le directory di descrizione delle pagine web.	N.E	RF1O 8.1.4
TS-RF1O 8.2	Verificare che Maap Framework crei automaticamente un account admin di default.	N.E	RF1O 8.2
TS-RF1F 8.3	Verificare che il framework MaaP permetta allo sviluppatore di definire un namespace per l'applicazione generata.	N.E	RF1F 8.3
TS-RF1O 9.1	Verificare che il DSL permetta allo sviluppatore di creare una pagina Collection-index.	N.E	RF1O 9.1
TS-RF1O 9.1.1	Verificare che il DSL deve permetta allo sviluppatore di poter definire una serie di attributi da visualizzare all'interno della pagina Collection-index.	N.E	RF1O 9.1.1
TS-RF1O 9.1.2	Viene verificato che il DSL permetta allo sviluppatore di poter definire un ordinamento di default (ordine alfanumerico) di visualizzazione dei document all'interno della pagina Collection-index.	N.E	RF1O 9.1.2
TS-RF1O 9.1.3	Verificare che il DSL permetta allo sviluppatore di poter definire un eventuale limite di elementi da visualizzare all'interno della pagina Collection-index.	N.E	RF1O 9.1.3



TS-RF1O 9.1.4	Viene verificato che il DSL permetta allo sviluppatore di poter definire quali attributi sono ordinabili all'interno della pagina Collection-index.	N.E	RF1O 9.1.4
TS-RF1O 9.1.5	Verificare che il DSL permetta allo sviluppatore di definire la funzione populate per far sì che una chiave riferisca ad un documento esterno.	N.E	RF1O 9.1.5
TS-RF1O 9.1.6	Verificare che il DSL permetta allo sviluppatore di definire delle query per creare la pagina Collection-index in base al risultato della loro estrazione.	N.E	RF1O 9.1.6
TS-RF1O 9.1.7	Viene verificato che il DSL permetta allo sviluppatore di definire delle trasformazioni sugli attributi da visualizzare.	N.E	RF1O 9.1.7
TS-RF1O 9.2	Viene verificato che il DSL permetta allo sviluppatore di creare una pagina Collection-show.	N.E	RF1O 9.2
TS-RF1O 9.2.1	Verificare che il DSL permetta allo sviluppatore di definire una serie di attributi visualizzabili all'interno della pagina Collection-show.	N.E	RF1O 9.2.1
TS-RF1O 9.2.2	Verificare che il DSL permetta allo sviluppatore la definizione degli attributi del Document come attributi innestati o array di Document tramite la funzione populate.	N.E	RF1O 9.2.2
TS-RF1O 9.2.3	Verificare che lo sviluppatore abbia la possibilità di personalizzare la show page definendone l'ordinamento degli attributi.	N.E	RF1O 9.2.3
TS-RF1O 9.2.4	Viene verificato che lo sviluppatore possa definire trasformazioni agli attributi per poi visualizzarli nella show-page.	N.E	RF1O 9.2.4
TS-RF1F 9.2.5	Verificare che lo sviluppatore possa personalizzare la show-page definendo delle operazioni personalizzate che l'utente potrà utilizzare tramite appositi pulsanti.	N.E	RF1F 9.2.5



TS-RF1O 9.3	Viene verificato che il framework MaaP permetta allo sviluppatore di cambiare il nome della Collection da visualizzare nel menu di navigazione.	N.E	RF1O 9.3
TS-RF1O 9.4	Verificare che il framework MaaP permetta allo sviluppatore di modificare l'ordine di visualizzazione della Collection nel menu di navigazione.	N.E	RF1O 9.4
TS-RS1F 10.1	Verificare che il sistema MaaS permetta allo sviluppatore di scrivere una Collection tramite editor di testo presente nella pagina web.	N.E	RS1F 10.1
TS-RS1F 10.2	Verificare che il sistema MaaS permetta all'utente di poter scrivere una Collection caricando un file prodotto dal framework MaaP.	N.E	RS1F 10.2
TS-RS1F 10.3	Verificare che il sistema MaaS permetta ad un utente non registrato di registrarsi al suo servizio.	N.E	RS1F 10.3
TS-RS1F 10.4	Verificare che il sistema MaaS assegni automaticamente un $namespace_G$ sul sistema al nuovo utente registrato.	N.E	RS1F 10.4
TS-RS1F 10.5	Verificare che il servizio MaaS visualizzi un messaggio d'errore nel caso in cui la registrazione fallisca a causa di credenziali già esistenti.	N.E	RS1F 10.5
TS-RS1F 10.6	Verificare che il servizio MaaS metta a disposizione di un utente non autenticato la possibilità di effettuare il login al sistema.	N.E	RS1F 10.6
TS-RS1F 10.7	Verificare che il servizio MaaS visualizzi un messaggio d'errore nel caso in cui l'utente non autenticato abbia inserito credenziali errate nel sistema di login.	N.E	RS1F 10.7
TS-RS1F 10.8	Verificare che il sistema MaaS permetta ad un utente non autenticato di modificare il proprio profilo.	N.E	RS1F 10.8
TS-RS1F 10.9	Verificare che il sistema MaaS permetta ad un utente non autenticato di eliminare il proprio account dal sistema.	N.E	RS1F 10.9



TS-RS1F 10.9.1	Verificare che il sistema MaaS provveda all'eliminazione dei file di configurazione associati all'utente rimosso dal sistema.	N.E	RS1F 10.9.1
TS-RS1F 10.10	Verificare che il sistema MaaS permetta allo sviluppatore di eliminare una Collection esistente.	N.E	RS1F 10.10
TS-RA1D 13.1	Verificare che l'utente possa modificare la password di accesso all'applicazione.	N.E	RA1D 13.1
TS-RF1O 14.1	Verificare che il framework MaaP renda possibile la configurazione dei database delle credenziali.	N.E	RF1O 14.1
TS-RF1O 14.2	Verificare che il framework MaaP renda possibile la configurazione dei database delle Collection.	N.E	RF1O 14.2
TS-RF1F 14.3	Verificare che il framework MaaP renda possibile la selezione di un namespace per un database se la funzione di $namespace_G$ è abilitata.	N.E	RF1F 14.3
TS-RA1F 15.1	Verificare che l'applicazione MaaP metta a disposizione dell'admin la visualizzazione degli indici in base alle query più richieste dall'applicazione.	N.E	RA1F 15.1
TS-RA1F 15.2	Verificare che l'applicazione MaaP permetta all'admin di aggiungere gli indici in base ai suggerimenti forniti.	N.E	RA1F 15.2
TS-RA1F 15.3	Verificare che l'applicazione MaaP permetta all'admin di rimuovere gli indici in base ai suggerimenti forniti.	N.E	RA1F 15.3
TS-RS1F 17	Verificare che Il sistema MaaS si accerti che documenti creati rispettano i vincoli del database.	N.E	RS1F 17
TS-RA1O 18	Verificare che il sistema metta a disposizione un validatore del codice DSL e visualizzi gli eventuali errori logici o di sintassi in un'apposita pagina.	N.E	RA1O 18
TS-RS1F 19	Verificare che il sistema MaaS salvi le pagine definite dagli utenti nel database e non su disco.	N.E	RS1F 19

Tabella A.1: Tracciamento Test di Sistema - Requisiti

A.3 Test di integrazione

I test di integrazione vanno a controllare il corretto funzionamento delle componenti descritti dalla progettazione ad alto livello. Si è scelto di utilizzare un approccio *top-down_G* ad eccezione del test TI 10 che viene eseguito con la metodologia *bottom-up_G*. Di seguito viene riportato un diagramma informale per chiarire l'albero dei test di integrazione.

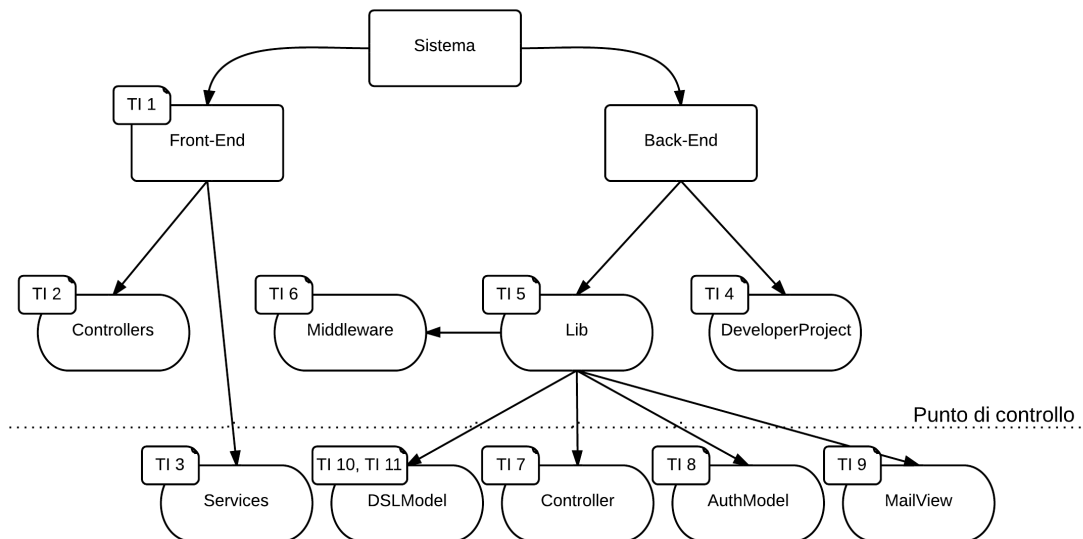


Figura 3: Sequenza d'integrazione delle componenti

Con la tecnica *top-down_G* le componenti di più alto livello sono testate non appena sono implementate. Le componenti del sottosistema che non sono ancora state sviluppate, vengono simulate dagli *stub_G*. Man mano che si procede con la codifica delle componenti di più basso livello, queste vengono integrate e viene eseguito il relativo test. Grazie all'integrazione incrementale delle componenti del sistema, è più semplice determinare quale componente crea problemi e le funzioni di più alto livello sono testate prima.

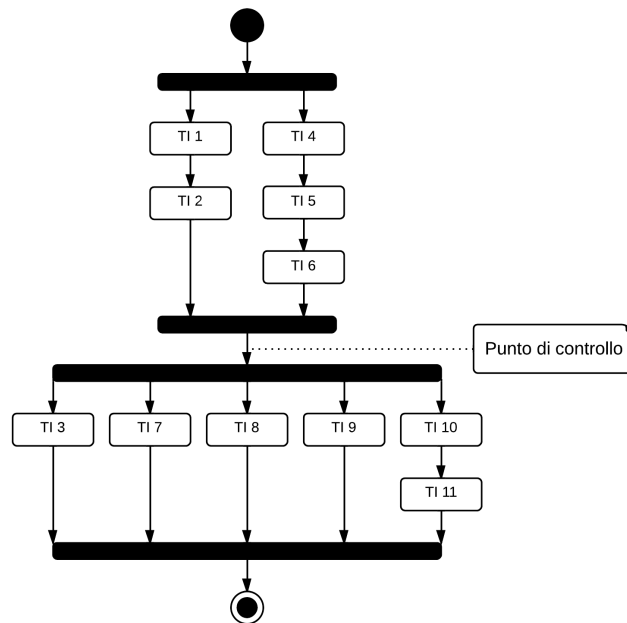


Figura 4: Diagramma di attività dei test

Test	Descrizione	Componenti aggiunte	Stato
TI 1	Si verifica che l'applicazione Web carichi correttamente le librerie JavaScript utilizzate.	Front-end	N.E.
TI 2	Si verifica che i controller si integrino correttamente nell'applicazione Web.	Front-end::Controller	N.E.
TI 3	Si verifica che i services permettano di interagire correttamente con il back-end.	Front-end::Services	N.E.
TI 4	Si verifica che il DeveloperProject avvii correttamente il server, fornendo in particolare i file statici del front-end.	Back-end::DeveloperProject	N.E.
TI 5	Si verifica che la libreria si integri correttamente con il <i>Node Package Manager_G</i> (npm) E che il suo script di installazione produca un DeveloperProject funzionante.	Back-end::Lib	N.E.
TI 6	Si verifica che il Middleware si integri correttamente nella gestione delle richieste che arrivano al server.	Back-end::Lib::Middleware	N.E.



TI 7	Si verifica che i controller si integrino correttamente nella gestione delle richieste che arrivano al server.	Back-end::Lib::Controller	N.E.
TI 8	Si verifica che l'AuthModel si integri correttamente con il Middleware della gestione dell'autenticazione.	Back-end::Lib::AuthModel	N.E.
TI 9	Si verifica che la MailView si integri correttamente con il Middleware della gestione dell'invio mail.	Back-end::Lib::MailView	N.E.
TI 10	Si verifica che le classi che compongono il DSLModel interagiscano correttamente tra loro.	Back-end::Lib::DSLModel	N.E.
TI 11	Si verifica che il DSLModel si integri correttamente con il funzionamento dell'applicazione.	Back-end::Lib::DSLModel	N.E.

Tabella A.2: Descrizione test d'Integrazione

A.4 Test di validazione

In questa sezione vengono elencati i test di validazione per verificare che il prodotto sia conforme alle attese. I test si svolgono seguendo e verificano tutti passi di cui si compongono.



Test di Validazione	Descrizione	Stato	Requisito
TV-RA1O 1	<p>L'utente non autenticato intende accedere all'applicazione, per farlo deve inserire le proprie credenziali composte da una email ed una password. All'utente è richiesto di:</p> <ul style="list-style-type: none">• Raggiungere la pagina di autenticazione;• Inserire la mail nel campo apposito;• Inserire la password;• Procedere con l'autenticazione.	N.E	RA1O 1
TV-RA1O 2	<p>L'utente intende recuperare la password d'accesso all'applicazione. All'utente è richiesto di:</p> <ul style="list-style-type: none">• Essere autenticato;• Raggiungere la pagina per il reset della password;• Richiedere il reset;• Raggiungere la casella email collegata all'account del sistema;• Seguire il link contenuto nella mail;• Compilare il form richiedente la nuova password;• Eseguire il Logout e autenticarsi con la nuova password.	N.E	RA1O 2
TV-RA1D 3	<p>L'utente autenticato può visualizzare la pagina di Dashboard nella quale potrà aver accesso ad esempio alla lista delle collection presenti e ad altre funzionalità disponibili. All'utente è richiesto di:</p> <ul style="list-style-type: none">• Accertarsi di essere autenticato;• Accedere alla pagina Dashboard tramite il menu di navigazione;	N.E	RA1D 3

TV-RA1O 4	<p>L'utente autenticato, selezionata una $Collection_G$, ne visualizza in forma tabellare tutti i documenti che contiene. Di questa collection può filtrarne i risultati visualizzabili, può eseguire tramite bottoni predisposti nella pagina azioni personalizzati e per ogni $Document_G$, selezionarlo e visualizzarne la show-page corrispondente.</p> <p>L'Admin ha i permessi per modificare un documento o eliminare un $Document_G$.</p> <p>All'utente è richiesto di:</p> <ul style="list-style-type: none"> • Essere autenticato; • Aprire la show-page relativa ad un Document; • Usare i filtri per filtrare la Collection • Eseguire un azione personalizzata, laddove presente; • Se admin, modificare un Document; • Se admin, eliminare un Document; 	N.E	RA1O 4
TV-RA1O 5	<p>L'utente visualizza la pagina show-page corrispondente ad un $Document_G$ selezionato visualizzandone gli attributi in forma tabellare.</p> <p>In questa pagina può aprire la show-page o l'index-page dell'array di $Document_G$ degli attributi innestati se presenti, eseguire un'operazione personalizzata se disponibile.</p> <p>L'Admin può eliminare il $Document_G$ a cui la show-page corrisponde o modificarlo. All'utente è richiesto di:</p> <p>Essere autenticato; Aprire la show-page degli attributi innestati; Aprire l'index-page dell'arra di Document; Eseguire, se presente, un operazione personalizzata; Se admin, modificare il Document; Se admin, eliminare il Document.</p>	N.E	RA1O 5



TV-RA1O 6	<p>L'Admin entra nella sua pagina di amministrazione nella quale visualizza una <i>Collection-Index_G</i> di tutti gli utenti registrati al sistema. All'utente è richiesto di:</p> <ul style="list-style-type: none">• Essere autenticato come admin;• Accedere alla pagina di creazione nuovi utenti;• Creare un nuovo utente;• Accedere alla pagina degli utenti registrati al sistema;• Visualizzare la pagina Collection-Show di un utente;	N.E	RA1O 6
TV-RF1O 8	<p>Lo sviluppatore deve poter creare un nuovo progetto tramite linea di comando.</p> <p>Allo sviluppatore è richiesto di:</p> <ul style="list-style-type: none">• Richiamare il comando di creazione di un nuovo progetto;• Passare come parametro il nome della directory che conterrà il progetto;• Verificare che siano state importate le librerie necessarie al corretto funzionamento del sistema;• Verificare che sia stato creato il file di configurazione di default dell'applicazione generata;• Verificare che sia stato creato il sistema di autenticazione per l'applicazione generata;• Verificare che siano state create le directory di descrizione delle pagine web;• Verificare che sia stato creato un account admin di default.	N.E	RF1O 8



TV-RF1O 9	<p>Lo sviluppatore deve poter configurare le Collection tramite il DSL di Maap Framework. All'utente è richiesto di:</p> <ul style="list-style-type: none">• creare una Collection-index tramite DSL;• creare una Collection-show tramite DSL;• modificare il nome della Collection;• modificare l'ordine di visualizzazione della Collection.	N.E	RF1O 9
TV-RS1F 10	<p>L'utente autenticato verifica che il $framework_G$ MaaP sia messo a disposizione dal sistema $MaaS_G$ come servizio Web.</p> <p>All'utente è richiesto di:</p> <ul style="list-style-type: none">• Accedere alla pagina di modifica del proprio profilo;• Modificare i dati associati al proprio profilo;• Verificare che i dati siano stati aggiornati;• Gestire i file di configurazione;• Eliminare il proprio account;• Verificare l'inaccessibilità al servizio tramite l'autenticazione con le credenziali associate all'account eliminato.	N.E	RS1F 10
TV-RA1D 11	<p>L'utente non autenticato deve potersi registrare all'applicazione MaaP.</p> <p>All'utente è richiesto di:</p> <ul style="list-style-type: none">• Inserire la mail nell'apposito campo di testo;• Inserire la password nell'apposito campo di testo;• Verificare che l'account sia stato registrato tramite l'autenticazione all'applicazione.	N.E	RA1D 11



TV-RA1D 12	<p>L'utente autenticato deve poter eseguire il logout dall'applicazione.</p> <p>All'utente è richiesto di:</p> <ul style="list-style-type: none">• Selezionare l'apposita opzione di logout;• Verificare di non essere più autenticato.	N.E	RA1D 12
TV-RA1D 13	<p>L'utente autenticato deve poter modificare le proprie credenziali d'accesso all'interno della propria pagina profilo.</p> <p>All'utente viene richiesto di:</p> <ul style="list-style-type: none">• Accedere alla propria pagina profilo;• Modificare la propria mail;• Modificare la propria password;• Eseguire il logout;• Autenticarsi con le nuove credenziali.	N.E	RA1D 13
TV-RF1O 14	<p>Lo sviluppatore deve poter configurare i database che compongono il sistema MaaP.</p> <p>Allo sviluppatore è richiesto di:</p> <ul style="list-style-type: none">• Configurare la connessione al database delle credenziali degli utenti;• Configurare il $namespace_G$ corrispondente, se la funzione di $namespace_G$ è abilitata;• Configurare la connessione al database delle $Collection_G$;• Configurare il $namespace_G$ corrispondente, se la funzione di $namespace_G$ è abilitata;• Selezionare un $namespace_G$ per il database da configurare, se la funzione di $namespace_G$ è abilitata.	N.E	RF1O 14



TV-RA1F 15	L'admin deve poter gestire gli indici da un'apposita pagina. All'admin è richiesto di: <ul style="list-style-type: none">• Accedere alla pagina di gestione degli indici;• Visualizzare i suggerimenti per la creazione degli indici;• Creare un indice;• Creare un indice da quelli suggeriti;• Eliminare un indice;• Eliminare un indice da quelli suggeriti.	N.E	RA1F 15
TV-RF1F 16	Lo sviluppatore deve poter abilitare i $namespace_G$ per l'applicazione creata. Allo sviluppatore è richiesto di: <ul style="list-style-type: none">• Attivare il $namespace_G$.	N.E	RF1F 16

Tabella A.3: Tracciamento Test di Validazione - Requisiti



B Resoconto delle attività di verifica

B.1 Riassunto delle attività di verifica

B.1.1 Revisione dei Requisiti

L'attività di verifica svolta dai *Verificatori* è avvenuta come determinato dal *Piano di Progetto v3.0.0* al termine della stesura di ogni documento previsto. La verifica svolta sui documenti è avvenuta seguendo le indicazioni delle *Norme di Progetto v3.0.0* e misurando le metriche indicate in 2.8.2. L'attività di *walkthrough* ha evidenziato una serie di anomalie, in questo modo è stato possibile stilare la lista di anomalie frequenti (vedi *Norme di Progetto v3.0.0*) che si potranno controllare tramite *Inspection*. Successivamente si è proceduto con le misurazioni delle metriche relative ai documenti. In questa revisione non è stato possibile valutare i processi poiché lo stato embrionale del team e impegni universitari sovrapposti non hanno permesso il rilevamento accurato di tutti i parametri necessari. Il gruppo ha in programma di colmare tale mancanza per la revisione successiva.

B.1.1.1 Miglioramenti A seguito delle attività di verifica e controllo è stato sottoposto un questionario ad ogni membro del gruppo che ha contribuito ad identificare le problematiche relative ai processi e a formulare proposte risolutive. Da queste idee sono nate diverse modifiche e miglioramenti ai documenti e in generale al nostro modo di lavorare. Seguendo questa linea abbiamo applicato coerentemente la politica di *plan-do-check-act*, utilissima per il miglioramento della qualità:

B.1.2 Revisione dei Requisiti

L'attività di verifica svolta dai *Verificatori* è avvenuta come determinato dal *Piano di Progetto v3.0.0* al termine della stesura di ogni documento previsto. La verifica svolta sui documenti è avvenuta seguendo le indicazioni delle *Norme di Progetto v3.0.0* e misurando le metriche indicate in 2.8.2. Successivamente si è proceduto con le misurazioni delle metriche relative ai documenti. Sono quindi state misurate le metriche sui processi per valutarne la bontà e fornire una base per la pianificazione dei cicli *PDCA_G*.



Problema	Possibile soluzione	Stato
Il dizionario personale di $Aspell_G$, essendo un file collaborativo compilato in automatico da tale $tool_G$, impone molto spesso attività manuali extra di gestione del $repository_G$, in particolare vanno risolti molti conflitti.	Uno script che ordina il file in questione dovrebbe diminuire i conflitti.	Da eseguire.
Contrassegnare le parole di glossario con il relativo tag_G è un'attività fortemente propensa a dimenticanze ed errori.	Uno script potrebbe contrassegnare le parole di glossario presenti nei documenti in automatico.	Da eseguire.
Scarsa frammentazione dei $task_G$	Incremento dell'utilizzo dello strumento di gestione dei processi.	Completato.
Mantenere la corrispondenza tra casi d'uso e relativi url_G dei diagrammi è un'operazione lunga e manuale.	Uno script può automatizzare tale attività.	Completato.
Difficoltà nell'uso dello strumento di controllo di versione.	I membri del gruppo devono eseguire attività extra di autoformazione sulla base del materiale messo a disposizione da alcuni membri.	Completato.

Tabella A.4: Problemi pre RR individuati e relative soluzioni.

B.2 Dettaglio delle verifiche tramite analisi

B.2.1 Analisi

B.2.1.1 Documenti Vengono qui riportati i valori dell'indice Gulpease per ogni documento durante l'analisi e relativo esito basato sui range stabiliti in 2.8.2.1.

Documento	Valore indice	Esito
<i>Analisi dei Requisiti v1.3.1</i>	45	superato
<i>Glossario v1.3.1</i>	63	superato
<i>Norme di Progetto v1.3.1</i>	47	superato
<i>Piano di Progetto v1.3.1</i>	51	superato
<i>Piano di Qualifica v1.3.1</i>	53	superato
<i>Studio di Fattibilità v1.3.1</i>	43	superato

Tabella A.5: Esiti verifica documenti, Analisi

B.2.2 Progettazione Architetture

B.2.2.1 Documenti Vengono qui riportati i valori dell'indice Gulpease per ogni documento durante la progettazione architetture e relativo esito basato sui range stabiliti in 2.8.2.1.

Documento	Valore indice	Esito
<i>Analisi dei Requisiti v3.0.0</i>	51	superato
<i>Glossario v3.0.0</i>	49	superato
<i>Norme di Progetto v3.0.0</i>	47	superato
<i>Piano di Progetto v3.0.0</i>	73	superato
<i>Piano di Qualifica v3.0.0</i>	50	superato
<i>Studio di Fattibilità v2.0.0</i>	44	superato
<i>Specifiche Tecniche v3.0.0</i>	58	superato

Tabella A.6: Esiti verifica documenti, Progettazione Architettuale

B.3 Dettaglio dell'esito delle revisioni

Lo sviluppo di questo progetto didattico si basa sull'attraversamento di quattro revisioni presiedute dal committente. Tre delle quattro revisioni produrranno delle segnalazioni degli errori riscontrati da parte del committente, deve seguire un report di come sono state risolte in ogni documento.

B.3.1 Revisione dei Requisiti

Per la revisione dei requisiti le segnalazioni da parte del committente sono state corrette:

- Norme di Progetto: il documento è stato riorganizzato per processi, attività, procedure, strumenti. Sono state aggiunte indicazioni sugli strumenti per la gestione del *repository_G* e le regole per la rotazione dei ruoli sono state definite in modo dettagliato;
- Analisi dei Requisiti: le *Norme di Progetto v3.0.0* descrivono la modalità di consegna che è stata ben definita che include la generazione dei nomi dei documenti con la relativa versione. Inoltre sono stati rivisti tutti i requisiti e casi d'uso segnalati dal committente;
- Piano di Progetto: l'Organigramma è stato spostato in appendice e sono stati rimossi i costi orari dei ruoli. Sono state ripartizionate le ore considerando attività di analisi successive al 2013-12-20 e la percentuale di ore di verifica è almeno del 30% del totale;
- Piano di Qualifica: la trattazione del *SEMAT_G* è stata spostata ed approfondita nel piano di progetto e le tecniche adottate sono state spostate nelle norme di progetto;
- Glossario: è stato creato l'indice del documento e ogni gruppo di lettera inizia su una pagina nuova.

C Qualità

La qualità perseguita nel presente documento si basa sugli standard ISO/IEC 15504 e ISO/IEC 9126 con l'obiettivo di approfondirne incrementalmente la copertura. Tutti i processi seguono il metodo $PDCA_c$ che prevede l'iterazione ripetuta tra i quattro stadi, incrementando di volta in volta quanto prodotto. La struttura di qualifica assicura un incremento della qualità ad ogni ciclo.

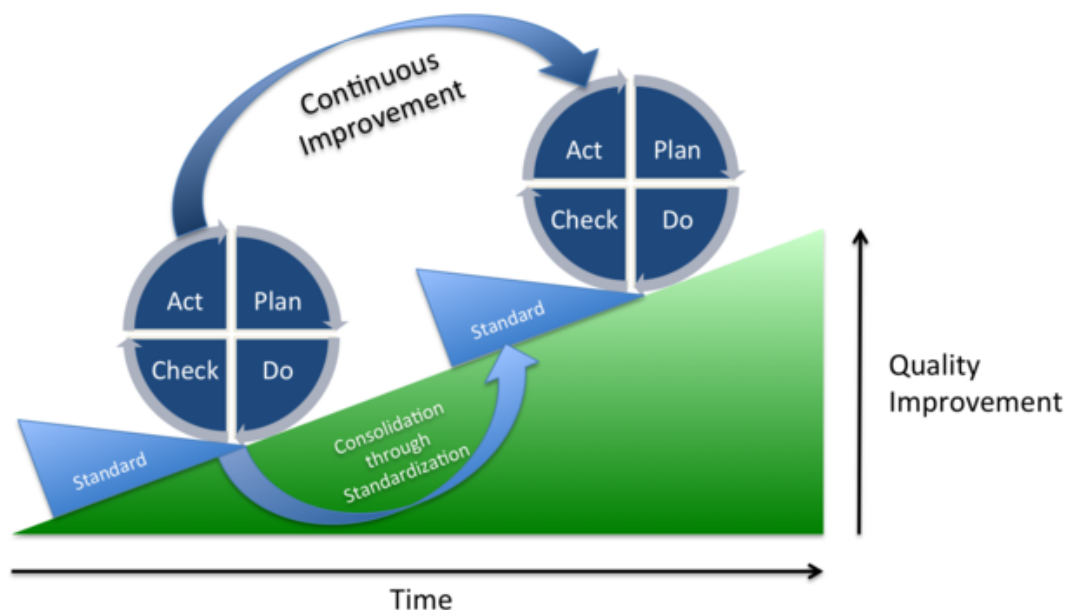


Figura 5: Continuous quality improvement with PDCA

1. **PLAN**: vengono stabiliti gli obiettivi e i processi necessari per raggiungere i risultati attesi, nel dettaglio:
 - Identificare il problema, o i processi da migliorare; Per descrivere il problema è necessario raccogliere i dati tramite misurazioni;
 - Analizzare il problema e individuare gli effetti negativi, definendo la loro importanza e le priorità di intervento;
 - Definire gli obiettivi di massima in modo chiaro e quantitativo, indicando i benefici ottenibili con il suo raggiungimento. Devono essere definiti anche i tempi, gli indicatori e gli strumenti di controllo.
2. **DO**: viene implementato il punto precedente, applicando le soluzioni individuate al problema;
3. **CHECK**: Verificare i risultati delle azioni intraprese, un confronto con i risultati attesi sarà il riscontro se quanto operato va nella direzione giusta. Vanno considerate metriche come

la *Schedule Variance* (vedi 2.8.1.1) e la completezza dei risultati attesi soddisfatti, vanno elaborati grafici e tabelle per avere una visione chiara di quanto rilevato. Se si è raggiunto l'obiettivo definito nello stadio di Plan, si può passare allo stadio di Act, altrimenti è necessario ripetere un nuovo ciclo PDCA sullo stesso problema, analizzando i vari stadi del ciclo precedente individuandone le cause del non raggiungimento dell'obiettivo stabilito;

4. **ACT:** La soluzione individuata viene standardizzata e tutti i membri del gruppo vengono informati e formati. Si potrà eseguire tramite riunioni o strumenti di messaggistica interna al gruppo. Terminato questo stadio si procederà con una nuova iterazione a partire dal punto 1.

Il ciclo $PDCA_G$ è stato attuato nel *Piano di Progetto v3.0.0*.

C.1 Qualità dei processi

Definita in ISO/IEC 15504 come $SPICE_G$, specifica come la qualità è collegata alla maturazione dei processi. Vengono individuati dei livelli di maturità al quale il fornitore può fare riferimento per determinare le proprie capacità organizzative. Vengono definiti:

- Dei **Modelli di riferimento** su:

- *dimensione del processo*;
- *livelli di capacità dei processi*:

5. ottimizzato
4. predicibile
3. stabilito
2. gestito
1. eseguito
0. incompleto

La capacità di un processo viene misurata tramite degli attributi che sono assimilabili alle metriche dei processi individuate in 2.8.1, in particolare la *Schedule Variance* permette di capire se un processo è incompleto o gestito; il gruppo giungerà a maturazione quando i processi diventeranno predicibili ossia quando la *Schedule Variance* subirà al più lievi oscillazioni;

- Delle **Stime** che si concretizzano in una struttura per la misurazione composta da:
 - I *processi* di misurazione, indicati nel *Piano di Progetto v3.0.0*;
 - Un *modello* per la misurazione identificabile in questo documento;
 - Gli *strumenti* utilizzati, specificati nelle *Norme di Progetto v3.0.0*.
- Le **Competenze e Qualifiche** di chi controlla; lo standard redige in modo rigoroso una serie di attività volte a formare chi opera l'attività di stesura del *Piano di Qualifica e Verifica*. Tali competenze sono assenti all'interno del gruppo e, considerato che effettuare una formazione in linea con quanto specificato dallo standard sarebbe impossibile, tutti i membri si impegnano a studiare ed applicare al meglio quanto descritto in questo documento.



C.2 Qualità del prodotto software

Specificata in ISO/IEC 9126 si suddivide in:

- **Quality model:** classifica la qualità del software in un set di caratteristiche che verranno approfondite nel corso del progetto:
 - Functionality: viene controllata grazie al tracciamento dei requisiti individuati ed analizzati e i componenti;
 - Reliability: viene dimostrata combinando i test;
 - Usability: viene controllata con i test di validazione, inoltre la stesura del manuale d'uso aiuterà a verificarne l'usabilità e ad intervenire laddove necessario;
 - Efficiency: combinando analisi statica e dinamica controlliamo che il prodotto sia efficiente;
 - Maintainability: viene realizzata con l'utilizzo di design pattern e la stesura di documentazione dettagliata;
 - Portability: essendo *MaaP_G* un applicazione Web non ci sono particolari problemi di portabilità per gli utenti.
- **External metrics:** sono le metriche rilevate tramite analisi dinamica, verranno specificate con il concretizzarsi della *Specifica Tecnica*;
- **Internal metrics:** sono le metriche rilevate in analisi statica specificate in 2.8;
- **Quality in use metrics:** si tratta di metriche rilevabili allo stato di prodotto *usabile* in condizioni reali, si rimanda la definizione di tale aspetto a quando verranno trattate le considerazioni sull'usabilità del prodotto in uno scenario di utilizzo reale, questo deve avvenire non oltre la *Progettazione di Dettaglio e Codifica*.



D PDCA

In questo capitolo verrà descritto come è stato applicato il modello $PDCA_G$ descritto nel *Piano di Qualifica v3.0.0*.

D.1 Revisione dei requisiti

In questo periodo è stata svolta un'attività di *walkthrough* non avendo i dati necessari per effettuare attività di *inspection*, come descritto nel *Piano di Qualifica v3.0.0*. Gli errori frequenti rilevati sono visionabili nelle *Norme di Progetto v3.0.0*.

Non è stato possibile eseguire nessun ciclo $PDCA_G$ in mancanza di misurazioni sui processi, non avendo quindi modo di pianificare processi per la qualità, ma è stato studiato e descritto nel *Piano di Qualifica v3.0.0* e verrà attuato dalla prossima *milestone_G*.

D.2 Revisione di progettazione

PLAN Al fine di valutare su quali processi pianificare dei processi di miglioramento sono state eseguite diverse misurazioni utilizzando le metriche per i processi descritte nel *Piano di Qualifica v3.0.0*.

I risultati ottenuti sono riportati nella seguente tabella:

Metrica	Valore indice	Esito
Produttività di documentazione	199	Superato
Impegno	0,71	Superato
Modifiche	23	Non superato

Tabella A.7: Risultati metriche per i processi, Revisione di progettazione

Analizzando tali dati si è deciso di pianificare le seguenti attività per il miglioramento della qualità dei processi:

Un numero troppo elevato di modifiche incide pesantemente sulla produttività. È necessario decrementare tale valore al fine di aumentare la produttività e di conseguenza diminuire i costi. Questo primo ciclo $PDCA_G$ si prefigge dunque l'obiettivo di portare entro un range di accettazione⁹ il numero di modifiche approvate.

Probabilmente un numero elevato di modifiche è causato dall'inesperienza del gruppo nel primo periodo, e ragionevolmente con l'aumentare delle conoscenze il numero di modifiche andrà calando di conseguenza.

In ogni caso, si pianifica di:

- Frammentare maggiormente i task assegnati in sotto-task;
- Specificare in modo esteso cosa prevede ogni singolo sotto-task, escludendo quindi dubbi che poi porteranno a successive richieste di modifica;

⁹Vedi *Piano di Qualifica v3.0.0*



- Creare la label “Domanda” nella sezione *issue_G* di *GitHub_G*, per permettere la richiesta di delucidazioni sullo svolgimento di *task_G* assegnati.

CHECK Al fine di valutare se le azioni pianificate hanno portato ad un miglioramento dei processi sono state eseguite le necessarie misurazioni.

I risultati ottenuti sono riportati nella seguente tabella:

Metrica	Valore indice	Esito
Produttività di documentazione	103	Superato
Impegno	2,69	Superato
Modifiche	18	Superato

Tabella A.8: Risultati metriche per i processi, Revisione di progettazione

Gli obiettivi posti nello stadio di pianificazione sono stati soddisfatti, si passerà dunque allo stadio di standardizzazione delle soluzioni applicate.