Sirius

Sequenziatore

Piano Di Qualifica

Versione 1.0.0

Ingegneria Del Software AA 2013-2014

Informazioni documento

Titolo documento:	Piano Di Qualifica
Data creazione:	21 Gennaio 2014
Versione attuale:	1.0.0
Utilizzo:	Interno
Nome file:	*PianoDiQualifica_v1.0.0.pdf*
Redazione:	Seresin Davide
Verifica:	Santangelo Davide
Approvazione:	Quaglio Davide
Distribuito da:	Sirius
Destinato a:	Prof. Vardanega Tullio
	Prof. Cardin Riccardo
	Zucchetti S.p.A.

Sommario

Il documento spiega in dettaglio la strategia di verifica adottata dal gruppo *Sirius* per lo sviluppo del progetto *Sequenziatore*.

Diario delle modifiche

Versione	Data	Autore	Ruolo	Descrizione
1.0.0	2014-03-05	*Quaglio Davide*	*Responsabile*	Approvazione del documento
0.2.0	2014-03-05	*Botter Marco*	*Verificatore*	Verifica delle aggiunte su resoconto dell'attivitá di verifica
0.1.1	2014-03-01	*Seresin Davide*	*Amministratore*	Aggiunta resoconto attivitá di verifica
0.1.0	2014-02-20	*Botter Marco*	*Verificatore*	Verifica del documento e appendice
0.0.3	2014-02-18	*Seresin Davide*	*Amministratore*	Aggiunta di informazioni dettagliate e appendice
0.0.2	2014-02-15	*Quaglio Davide*	*Verificatore*	Verificato scheletro e bozza documento
0.0.1	2014-02-12	*Seresin Davide*	*Amministratore*	Creato lo scheletro del documento

Indice

1 Introduzione

1.1 Scopo del Prodotto

Lo scopo del progetto *Sequenziatore*, é di fornire un servizio di gestione di processi definiti da una serie di passi da eseguirsi in sequenza o senza un ordine predefinito, utilizzabile da dispositivi mobili di tipo smaptphone o tablet.

1.2 Glossario

Al fine di rendere piú leggibile e comprensibile i documenti, i termini tecnici, di dominio, gli acronimi e le parole che necessitano di essere chiarite, sono riportate nel documento *Glossario_v1.0.0.pdf*.
Ogni occorrenza di vocaboli presenti nel *Glossario* deve essere seguita da una "G" maiuscola in pedice.

1.3 Riferimenti

1.3.1 Normativi

- ISO/IEC Standard 12207:1995;
- ISO/IEC 9126;
- IEEE Std 730TM-2002 (revision of IEEE Std 730-1998) - Standard for Software Quality Assurance Plans;
- **Norme di progetto:** Norme di progetto v1.0.0;
- Capitolato d'appalto C4: *sequenziatore*.

1.3.2 Informativi

- Informazioni sul sito del docente;
- Software Engineering (9th edition) Ian Sommerville Pearson Education Addison-Wesley;
- Ciclo di Deming - Estratto da Software Engineering (9th edition) Ian Sommerville Pearson Education Addison-Wesley;
- Capability Maturity Model Integration (CMMI) - Estratto da Software Engineering (9th edition) Ian Sommerville Pearson Education Addison-Wesley;
- SWEBOK cap.11 - *Software Quality*;
- **Piano di progetto:** Piano Di Progetto v1.0.0;

- Indice di Gulpease.

1.4 Scopo del documento

Il documento si prefigge di illustrare la strategia complessiva di verifica e validazione proposta dal team *Sirius* per pervenire al collaudo del sistema con la massima efficienza$_G$ ed efficacia$_G$; inoltre questo documento pretende di essere a volte utilizzato come manuale della qualitá per il gruppo *Sirius*; definendo gli obiettivi di qualitá intesa come il rispetto dei requisiti e prestazioni enunciati esplicitamente, la conformitá agli standard di sviluppo esplicitamente documentati e le caratteristiche implicite che si aspetta da un prodotto software. Garantendo in particolar modo:

- La correttezza del prodotto;
- La verifica continua sulle attivitá svolte;
- Il soddisfacimento del cliente.

2 Visione generale della strategia di verifica

2.1 Introduzione

Il team *Sirius* ha deciso di porre al centro di ogni fase l'attivitá di verifica in quanto essa certifica la qualitá del prodotto. L'attivitá di verifica sará continua in tutte le fasi del progetto.

2.2 Organizzazione

Il processo di qualifica accompagnerá tutte le fasi di ciclo di vita del software. Ogni procedura di verifica sará schedulata attraverso appositi strumenti e i risultati saranno analizzati in questo documento. Tramite il diario delle modifiche é possibile tenere traccia dell'attivitá di verifica effettuata ed operare delle verifiche circoscritte ai soli cambiamenti. *Sirius* ha deciso di adottare una politica per lo sviluppo del progetto a ciclo di vita incrementale; tale scelta ha portato ad organizzare le attività di verifica nei periodi temporali antecedenti le revisioni.

2.3 Pianificazione strategica temporale

Al fine di rispettare in modo ristretto le scadenze citate di seguito e spiegate in modo approfondito in Piano Di Progetto v1.0.0, *Sirius* ha deciso di pianificare in modo approfondito e sistematico l'attivitá di verifica. Facendo in modo di rilevare e risolvere nel piú breve tempo possibile gli errori che vengono rilevati per evitare che questi possano creare maggiori problematiche nell' avanzamento del prodotto software. Le scadenze sono di due tipi:

- Scadenze formali
  - Revisione dei requisiti (RR): *2014-03-05*;
  - Revisione di accettazione (RA): *2014-07-18*.
- Scadenze di avanzamento
  - Revisione di progettazione (RP): *2014-04-02*;
  - Revisione di qualifica (RQ): *2014-07-02*.

2.4 Descrizione della procedura di pianificazione adottata

Per gestire al meglio la pianificazione, *Sirius* prevede l'attivitá di verifica in tutti e quattro i periodi di avanzamento del prodotto, che sono paralleli alle scadenze definite in Piano Di Progetto v1.0.0.

- **Analisi**: in questa prima fase il compito del *Verificatore* é innanzitutto relativo alla documentazione e alla correttezza del tracciamento dei requisiti. Ogni documento che servirá per la consegna della RR, una volta ultimata la fase di redazione, verrá verificato in modo definitivo seguendo la procedura cosí definita:

  1. Verrá controllata la correttezza dei contenuti rispetto alle aspettative del documento tramite una rilettura accurata;
  2. Verrá controllata la correttezza grammaticale;
  3. Verrá controllato che il documento rispetti le norme definite in Norme di progetto v1.0.0 tramite la lista di controllo presente in tale documento.
  4. Verrá verificato che ogni requisito funzionale rilevato abbia una corrispondenza in almeno un caso d'uso e che questo sia tracciato tramite il software di tracciamento che *Sirius* ha deciso di utilizzare;
  5. Verrá verificato che ogni requisito di vincolo e di qualitá sia tracciato tramite il software di tracciamento che *Sirius* ha deciso di utilizzare.

- **Progettazione**: il *Verificatore* ha l'importante compito di controllare il soddisfacimento dei requisiti indicati in fase di analisi. Inoltre, si devono verificare che i processi che portano all'incremento dei documenti redatti nel precedente periodo siano conformi alle procedure e regole descritte in Norme di progetto v1.0.0;

- **Programmazione**: il *Verificatore* provvederá a controlli periodici e pianificati di porzioni di codice, inizialmente di tipo statico per poi passare a dei controlli di tipo dinamico per valutare la correttezza del software;

- **Collaudo**: in questa fase le verifiche saranno esclusivamente di tipo dinamico per garantire che il prodotto risponda a tutti i requisiti indicati e a tutte le richieste del committente: sia implicite che esplicite.

Per i diversi periodi l'attivitá di verifica sará diversa in base alla tipologia di lavoro svolta. Essa verterá sulla verifica della documentazione all'analisi del prodotto software garantendo che il risultato sia efficace$_G$ rispetto alla procedura analizzata, non perdendo di efficienza$_G$ contrattuale.

2.5 Slack

Sirius, conscio della poca esperienza nella pianificazione e gestione di progetti di questo tipo, ha deciso di inserire degli *slack*$_G$ temporale durante la pianificazione delle attivitá. Tale scelta é approfondita in Norme di progetto v1.0.0 che ne definisce la quantiá e in Piano Di Progetto v1.0.0 che ne analizza le motivazioni. L'aggiunta di slack temporali, oltre a portare al progetto una pianificazione piú precisa, comporta un aumento dei costi che é peró commisurato all'aumento della qualitá finale.

2.6 Obiettivi

2.6.1 Qualitá di processo

Al fine di garantire la qualitá di prodotto é necessario ricercare la qualitá dei processi che lo definiscono. Per questo *Sirius* ha deciso di adottare lo standard ISO/IEC 15504 denominato SPICE il quale fornisce le indicazioni necessarie a valutare l'idoneitá dei processi attualmente in uso. Per applicare correttamente questo modello, ed adattarlo alla gestione attuata dal gruppo di lavoro, si é deciso di utilizzare il ciclo di Deming o PDCA: il quale definisce una metodologia di controllo dei processi durante il loro ciclo di vita permettendo, inoltre, di migliorare in modo continuo la qualitá. Tramite al principio PDCA, descritto nella sezione A.1, é garantito un miglioramento continuo dei processi e quindi della qualitá degli stessi: come diretta conseguenza si otterrá il miglioramento qualitativo del prodotto risultante. Per attenere questo é necessario che il processo sia in controllo e quindi:

- Effettuare una dettagliata pianificazione dei processi;
- Pianificare anche il numero di risorse da utilizzare;

La qualitá dei processi viene monitorata anche tramite la qualitá del prodotto: infatti un pprodotto di bassa qualitá indica un processo da migliorare.

2.6.2 Qualitá di prodotto

Al fine di aumentare il valore del prodotto e di garantire il corretto funzionamento dello stesso é necessario fissare degli obiettivi e garantire che questi vengano effettivamente rispettati. Lo standard ISO/IEC 9126 é stato redatto con lo scopo di descrivere questi obiettivi e di delineare delle metriche capaci di misurare il raggiungimento degli stesso. Ognuno di questi obiettivi verrá valutato al termine del processo di verifica per avere dei risultati da analizzare. La valutazione dei risultati ottenuti é molto importante, soprattutto nei primi momenti di creazione del gruppo, in quanto questo deve essere un punto di partenza per modificare, creare o ridefinire le procedure in modo da evitare il ripercuotersi sistematico degli errori. In particolare *Sirius* si impegna a garantire la qualitá di prodotto perseguendo le seguenti caratteristiche:

- **Funzionalitá** L'applicazione prodotta deve soddisfare tutti i requisiti individuati nell'Analisi dei Requisiti nel modo piú completo ed economico possibile, garantendo la sicurezza del prodotto e dei suoi componenti, e adeguandosi alle norme e alle prescrizioni imposte. *Sirius*ha stabilito che la soglia di sufficienza sia il soddisfacimento di tutti i requisiti obbligatori.
- **Affidabilitá** L'applicazione deve dimostrarsi robusta, di facile ripristino e recupero in caso di errori, e aderire alle norme e alle prescrizioni stabilite. Per questo

*Sirius*effettuerá i test di sistema in modo approfondito al fine di testare il maggior numero di casistiche di esecuzione.

- **Usabilitá** L'applicazione deve risultare comprensibile, facilmente apprendibile e soprattutto aderire a norme e prescrizioni per garantire facilitá d'uso e soddisfacimento delle necessitá dell'utente.
- **Efficienza** L'applicazione deve fornire tutte le funzionalitá nel minor tempo possibile e con il minimo utilizzo di risorse. Per questo si faranno dei test di sistema nei primi periodi di sviluppo software al fine di ricercare la soluzione che permetta di ottenere risultati dalla richiesta di funzionalitá nel minor tempo possibile.
- **Manutenibilitá** L'applicazione deve essere analizzabile, facilmente modificabile e verificabile. Per questo é stata definita una metrica indicata di seguito.
- **Portabilitá** L'applicazione deve essere adattabile e compatibile con ambienti d'uso diversi quindi validata da strumenti forniti dal W3C.
- **Semplicitá**: realizzazione del prodotto nella maniera pi‘ semplice possibile, ma non semplicistica;
- **Incapsulamento**: il codice deve avere visibilitá minima e permettere un utilizzo dall'esterno solamente mediante interfacce; ció aumenta la manutenibilitá e la possibilitá di riuso del codice;
- **Coesione**: le funzionalitá che concorrono allo stesso fine devono risiedere nello stesso componente; favorisce semplicitá, manutenibilitá, riusabilitá e riduce l'indice di dipendenza.

Al fine di controllare la qualitá di prodotto raggiunta *Sirius*ha adottato le seguenti procedure:

- ***Quality assurance***: insieme di attivitá atte a garantire il raggiungimento degli obiettivi di qualitá;
- **Verifica**: determina se l'output di una fase é **consistente**, **completo** e **corretto**. Il processo di verifica é eseguito in modo uniforme e durante tutto il periodo di realizzazione del prodotto. I risultati della verifica sono analizzati di volta in volta e riportati in seguito al documento;
- **Validazione**: é si intende la conferma in modo oggettivo che il sistema risponde ai requisiti.

2.7 Risurse umane e responsabilitá

Il responsabile delle relazioni con il committente é il *Responsabile di Progetto* il quale é responsabile anche dell'operato dei *Verificatori*. Mentre i *Verificatori* sono a loro volta responsabili delle attivitá di verifica e delle procedure di controllo qualitá . Il *Verificatore*, inoltre, avrá un ruolo attivo in tutto il ciclo di vita del prodotto software, ma esso dovrá evitare di influire in modo troppo pesante sui processi software per evitare un rapporto costi/benefici troppo elevato. Compito dell' *Amministratore* á quello di supporto a tutte le attivitá fornendo una solida infrastruttura software anche per il processo di verifica in ogni fase lavorativa. Per una descrizione piú approfondita di ruoli e responsabilitá si rimanda a Norme di progetto v1.0.0.

2.8 Risorse software

Al fine di effettuare la fase di verifica e validazione nel podo più sistematico possibile sono stati messi a disposizione di tutti i *Verificatori* dall' *Amministratore* un pacchetto di prodotti software il piú specifico possibile rispetto alle esigenze del team. Inoltre, é sempre compito dell' *Amministratore* formare ogni verificatore all'utilizzo dei prodotti che permettano la verifica, evidenziando, se richiesto le funzionalitá non utilizzate per ogni prodotto. *Sirius* ha deciso di adottare questo tipo di formazione per valorizzare il lavoro di chi effettivamente utilizza i prodotti di verifica, dando la possibilitá che proprio da queste figure nascano idee e proposte di miglioramento che saranno poi valutate dal *Responsabile di progetto* congiuntamente all'*Amministratore*. Gli strumenti necessari al raggiungimento degli obiettivi definiti é indicato in Norme di progetto v1.0.0, che ne definiscono anche le specifiche tecniche e l'utilizzo.

2.9 Tecniche di analisi statica

L'analisi statica é una tecnica di analisi applicabile sia alla documentazione che al codice e permette di effettuare la verifica di quanto prodotto individuando errori ed anomalie. Essa può essere svolta in due modi diversi ma complementari tra di loro in quanto per utilizzare *inspection* bisogna prima aver effettuato *walkthrough*

2.9.1 Inspection

Questa tecnica, di analisi statica, consiste nella verifica di sezioni ben definite di un documento o del codice. Questo tipo di controlli per i documenti sono usualmente definiti tramite una lista di controllo (checklist) redatta anticipatamente rispetto all'attivitá di verifica da intraprendere. Per la verifica dei documenti, la lista di controllo é stata elaborata a seguito di analisi eseguite tramite *walkthrough*, ed evidenziando gli errori più ricorrenti riscontrati. *Inspection* é una strategia rapida in quanto permette l'analisi

di alcuni parti ritenute critiche nella checklist senza bisogno di una lettura integrale di documento o di tutto il codice in oggetto.

2.9.2 Walkthrough

Walkthrough é una tecnica di analisi statica che consiste nella lettura critica a largo raggio di tutto il documento. In questa tipologia di analisi il *Verificatore* utilizza molto tempo per la lettura e correzione del documento o codice. Questa tecnica viene di solito utilizzata nella prima parte dello sviluppo di progetti in quanto, la poca esperienza del *Verificatore* non permette un'altro tipo di verifica. Al termine di questo primo set di analisi *walkthrough* viene usualmente definita una lista di controllo che permetta di ricercare in primo luogo gli errori piú ricorrenti, e maggiormente riscontrati. *Walkthrough* é un'attivitá onerosa e collaborativa che richiede l'intervento di piú persone per essere efficente ed efficace

2.10 Tecniche di analisi dinamica

L'analisi dinamica si applica solamente al prodotto software e consiste nell'esecuzione del codice mediante l'uso di test predisposti per verificarne il funzionamento o rilevare possibili difetti di implementazione eseguendo tutto o solo una parte del codice. La **ripetibilitá** del test é una caratteristica fondamentale per questo tipo di test, in quanto dichiara che il codice con un certo *input* produce sempre lo stesso *output* su uno specifico ambiente. In questo modo si é in grado di riscontrare problemi e verificare la correttezza del prodotto. Per questo *Sirius* ha deciso di definire a priori le seguenti caratteristiche:

- **Ambiente**: sistema *hardware* e quello *software* sui quali é stato pianificato l'utilizzo del prodotto, di essi si deve definire uno stato iniziale dal quale poter iniziare ad eseguire i test;
- **Specifica di *input***: definire quali sono gli *input* e quali devono essere gli *output* attesi;
- **Procedure**: definire quali devono essere i test ed in che ordine devono essere analizzati i risultati ottenuti.

Di seguito sono definiti cinque diversi tipi di test.

2.10.1 Test di unitá

Per test di unitá si intende la verifica di ogni singola unitá di prodotto software tramite l'utilizzo di stub_G, driver_G e logger_G. Per unitá si intende la piú piccola porzione di codice che é utile verificare singolarmente e che viene prodotta da un unico programmatore. Tramite questo tipo di test si vogliono testare i vari le unitá per rilevare errori di implementazione da parte dei programmatori.

2.10.2 Test di integrazione

I test di integrazione prevedono la verifica dei componenti del sistema che vengono aggiunti incrementando il prodotto di origine e si prefigge quindi di analizzare la combinazione di due o più unitá software che hanno quindi superato i test di unitá. Questa tecnica di verifica serve ad individuare errori residui nella programmazione dei singoli moduli: come modifiche delle interfacce e comportamenti inaspettati di componenti software di parti terze e che pregiudicherebbero la validitá del prodotto. Per effettuare tali test puó essere necessario l'aggiunta di componenti software fittizzie e non ancora implementate al fine di non pregiudicare negativamente l'esito dell'analisi.

2.10.3 Test di sistema

Consiste nella validazione del sistema attraverso la verifica della copertura di tutti i requisiti obbligatori individuati in Analisi dei requisiti v1.0.0, e tracciati grazie allo strumento messo a punto da *Sirius*;

2.10.4 Test di regressione

I test di regressione vengono eseguiti quando si apportano delle modifiche a parte del software e questi consistono nella riesecuzione dei test riguardanti le i componenti che hanno subito modifiche e che precedentemente non erano soggetti ad errori. Tale operazione viene aiutata dal tracciamento, che permette di individuare e ripetere facilmente i test di unitá, integrazione ed eventualmente di sistema che sono stati potenzialmente influenzati dalle modifiche.

2.10.5 Test di accettazione

Si tratta del collaudo del prodotto software sotto il controllo del proponente. Se il collaudo viene superato in modo positivo, il sistema viene rilasciato e la commessa si conclude.

2.11 Misure e metriche per l'accettazione

I dati rilevati dal processo di verifica devono essere analizzati tramite precise metriche. Con questo termine si intende l'insieme di parametri misurabili su un processo. Qualora le metriche definite in questo documento siano approssimative e/o ambigue, queste dovranno essere ridefinite in modo specifico e seguiranno in modo incrementale il ciclo di vita del prodotto. Per ogni metrica sono definiti due diverse tipologie di intervalli:

- Accettazione: intervallo di valori affinché il prodotto sia accettato;

- Ottimale: all'interno dell'intervallo di accettazione viene definito l'intervallo ottimale all'interno del quale si dovrebbe posizionare la misurazione effettuata. Tale intervallo non é vincolante ma fortemente consigliato.

2.12 Metriche per i processi

Le metriche dei processi ne stabiliscono la qualitá che é definita come connubio di *capability*$_G$, *maturity*$_G$ e i miglioramenti. Queste caratteristiche di qualitá si possono individuare in tre classi di misure di processo:

- **Tempo**: il tempo richiesto per il completamento di un particolare processo;
- **Risorse**: le risorse richieste per un particolare processo, in genere vengono definite risorse-uomo, per le risorse software si fa riferimento a Norme di progetto v1.0.0;
- **Occorrenze**: il numero di volte che capita un particolare evento, che puó essere il numero di difetti scoperti durante l'attivitá di verifica.

Per rilevare questi dati *Sirius* ha deciso di utilizzare, per il controllo dei processi, indici che valutano i tempi e i costi del processo. La scelta di queste metriche é dettata anche dal loro possibile utilizzo anche durante lo svolgimento del processo, per capire in modo semplice se lo stato del processo é conforme a quanto pianificato, mantenendo quindi il processo in controllo. In Piano Di Progetto v1.0.0 viene specificato come sono stati pianificati questi indici nello stato di avanzamento.

2.12.1 (SV) Schedule Variance

Indica se si é in linea, in anticipo o in ritardo rispetto alla pianificazione temporale delle attività citata in Piano Di Progetto v1.0.0. È un indicatore di efficacia temporale e per questo *Sirius*ha deciso di esprimerlo in ore. Se SV > 0 significa che il gruppo di lavoro sta producendo con maggior velocitá rispetto a quanto pianificato, viceversa se negativo.
Parametri utilizzati:

- Accettazione: [>-(ore preventivo fase x5%)];
- Ottimale: [>0].

2.12.2 (BV) Budget Variance

Indica se allo stato attuale si é speso piú o meno rispetto a quanto pianificato. È un indicatore che ha valore contabile e finanziario per questo é espresso in euro. Se BV

> 0 significa che l'attuazione del progetto sta consumando il proprio budget con minor velocitá rispetto a quanto pianificato, viceversa se negativo. **Parametri utilizzati:**

- Accettazione: [>-(costo preventivo fase x10%)];
- Ottimale: [>0].

2.13 Metriche per i documenti

Come metrica per la verifica dei documenti *Sirius*ha deciso di utilizzare l'indice di leggibilitá. Vi sono a disposizione molti indici di leggibilitá, ma i più importanti sono per la lingua inglese. Si é deciso quindi di adottare un indice di leggibilitá per la lingua italiana. L'indice *Gulpease* é un indice di leggibilitá di un testo tarato sulla lingua italiana. Rispetto ad altri indici, esso ha il vantaggio di utilizzare la lunghezza delle parole in lettere anziché in sillabe, semplificandone il calcolo automatico. Permette di misurare la complessitá dello stile di scrittura di un documento. L'indice viene calcolato utilizzando la formula citata nelle Norme di progetto v1.0.0 . I risultati sono compresi tra 0 e 100, dove il valore 100 indica la leggibilitá più alta e 0 la leggibilitá piú 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:

- Accettazione: [40-100];
- Ottimale: [50-100].

2.14 Metriche per il software

Al fine di perseguire gli obiettivi qualitativi dichiarati sopra é necessario definire delle metriche per fare in modo che questi obiettivi siano misurabili. Questa sezione, però, é da intendersi come modificabile nell'arco dello svolgimento del progetto in quanto sará a discrezione del tema utilizzare le metriche che sono piú rappresentative.

2.14.1 Complessitá ciclomatica

Pensata da T.J. McCabe é utilizzata per misurare la complessitá per funzioni, moduli, metodi o classi di un programma. Misura direttamente il numero di cammini linearmente indipendenti attraverso il grafo di controllo di flusso. Alti valori di complessitá

ciclomatica indicano una ridotta manutenibilitá del codice. Al contrario, valori bassi potrebbero determinare una scarsa efficienza dei metodi. Questo parametro é inoltre un indice del carico di lavoro richiesto dal *testing*. Indicativamente un modulo con complessitá ciclomatica piú bassa richiede meno test di uno con complessitá piú elevata.
Parametri utilizzati:

- Accettazione: [1-15];
- Ottimale: [1-10].

2.14.2 Numero livelli di annidamento

Rappresenta il numero di livelli di annidamento quindi l'inserimento di una struttura di controllo all'interno di un'altra. Un elevato valore comporta un'alta complessitá e un basso livello di astrazione del codice.
Parametri utilizzati:

- Accettazione: [1-6];
- Ottimale: [1-3].

2.14.3 Attributi per classe

Un elevato numero di attributi per classe puó rappresentare la necessitá di suddividere la classe in piú classi, possibilmente utilizzando la tecnica dell'incapsulamento, e può inoltre rappresentare un possibile errore di progettazione.
Parametri utilizzati:

- Accettazione: [0-16];
- Ottimale: [3-8].

2.14.4 Numero di parametri per metodo

Un elevato numero di parametri potrebbe richiedere di ridurre le funzionalitá del metodo o provvedere ad una nuova progettazione dello stesso.
Parametri utilizzati:

- Accettazione: [0-8];
- Ottimale: [0-4].

2.14.5 Linee di codice per linee di commento

Indica il rapporto tra linee di codice e linee di commento: questo parametro é fondamentale per valutare la manutenibilitá del codice prodotto, nonché del possibile riuso.
Parametri utilizzati:

- Accettazione: [¿0,25];
- Ottimale: [¿0.30].

2.14.6 Flusso di informazioni

fan in e fan out DA TOGLIERE
Parametri utilizzati:

- Accettazione: [40-100];
- Ottimale: [50-100].

2.14.7 Accoppiamento

accoppiamento afferente e accoppiamento efferente DA TOGLIERE
Parametri utilizzati:

- Accettazione: [40-100];
- Ottimale: [50-100].

2.14.8 Copertura del codice

Indica la percentuale di istruzione che vengono eseguite durante i test. Maggiore é la percentuale e piú probabilitá si hanno di rilevare minori errori nei prodotto. Tale valore puó essere abbassato tramite l'utilizzo di metodi molto semplici che non richiedono test.
Parametri utilizzati:

- Accettazione: [42%-100%];
- Ottimale: [65%-100%].

3 Gestione amministrativa della revisione

3.1 Comunicazione e risoluzione anomalie

Un'anomalia consiste in una deviazione del prodotto dalle aspettative prefissate. Per la gestione e risoluzione di anomalie ci si affida allo strumento di *ticketing* adottato da *Sirius*, e normato in Norme di progetto v1.0.0. Il *Verificatore*, per ogni anomalia riscontrata, dovrá aprire un nuovo ticket indirizzato al *Responsabile di Progetto*, il quale, dopo aver valutato l'impatto costi/benefici lo approverá e aprirá in ticket per il *Programmatore* che ha sviluppato quella parte di software o redatto il documento. Per la procedura di creazione del ticket si rimanda a Norme di progetto v1.0.0.

3.2 Trattamento delle discrepanze

Una discrepanza é un errore di coerenza tra il prodotto realizzato e quello atteso. *Sirius* interpreta la discrepanza come una forma di anomalia non grave, e per questo verrá trattata come tale.

3.3 Procedure di controllo di qualità di processo

Le procedure di controllo qualitá di processo si basano sul ciclo di Deming, che lo arricchisce. Fissare gli obiettivi, é la prima attivitá del ciclo di Deming di processo; tutte le successive attivitá devono mutare nel tempo per fare in modo che possano essere migliorate in modo continuativo. I processi saranno pianificati dettagliatamente e ogni pianificazione prevederá dei valori attesi dallo stesso, questi saranno confrontati con i risultati ottenuti alla terminazione del processo e analizzati. Se l'analisi di tali misure evidenzia valori che si discostano, in modo negativo, dal valore atteso, questo sará indice di un'opportunitá di miglioramento. Per ognuno di questi valori si ricercheranno le cause e si definiranno specifiche soluzioni intervenendo sul processo stesso ed eventualmente anche sul valore definito nella pianificazione iniziale. *Sirius* ha definito le principali misurazioni di processo:

- *Lead time* preventivato e *lead time* a consuntivo;
- Risorse utilizzate durante il processo;
- Cicli di processo;
- Attinenza alla pianificazione iniziale;
- Soddisfacimento dei requisiti richiesti.

Se non vengono rilevati problemi relativi ad un processo, è possibile aumentare l'efficenza del processo studiando tecniche migliorative che permettano di abbassare il *lead time* o il numero di risorse impiegate, garantendo sempre che il prodotto finale abbia

un elevato grado di soddisfacimento dei requisiti richiesti. Ad ogni modo, ogni singola misurazione puó essere utilizzata per una piú specifica pianificazione nelle successive esecuzioni di processo.

4 Pianificazione dei test

4.1 Test di sistema

4.1.1 Descrizione dei test di sistema

4.2 Test di integrazione

4.2.1 Descrizione dei test di integrazione

4.2.2 Tracciamento componenti-test di integrazione

4.3 Test di validazione

4.3.1 Test TV1

4.3.2 Test TV2

4.3.3 Test TV2

4.3.4 Test TV2

4.3.5 Test TV2

4.3.6 Test TV2

4.3.7 Test TV2

4.3.8 Test TV2

5 Resoconto attivitá di verifica

5.1 Riassunto dell'attivitá di verifica su RR

Sirius, ha deciso di valorizzare soprattuto i requisiti desiderabili per fare in modo di tenere alto l'indice di efficienza$_G$. Valutando, durante lo stato di avanzamento, quali di questi requisiti saranno successivamente sviluppati nella realizzazione del prodotto software. Questo tipo di scelte risultano infatti difficoltose allo stato attuale in quanto una pianificazione approfondita necessiterebbe di *lead time* $_G$ precisi che non sono al momento tra le conoscenze dei componenti del gruppo. Nel periodo di tempo che ha portato *Sirius* alla consegna di questa revisione sono stati verificati i **documenti** ed i **processi**.

I **documenti** sono stati verificati anche durante le operazioni di redazione per portare a conoscenza dei contenuti tutti i componenti del gruppo di lavoro. L'analisi statica, in primo luogo utilizzando la tecnica del *walkthrough*, ha portato alla redazione di una lista di controllo che verrá poi incrementata ed utilizzata nell'analisi finale del documento prima di procedere alla consegna. Una volta rilevati gli errori questi sono stati notificati al redattore che ha proceduto alla correzione, evidenziando gli errori frequenti che sono stati utilizzati per migliorare il processo di verifica. *Sirius* adotta il ciclo PDCA per rendere piú efficente$_G$ ed efficace$_G$ nel tempo il processo di verifica.

L'attivitá di verifica, inoltre, utilizzando la tecnica *inspection* é stata utilizzata principalmente per la verifica dei grafici dei casi d'uso. Per verificare la correttezza dei requisiti richiesti e la successiva completezza ci si é affidati ad un particolare strumento di tracciamento definito in Norme di progetto v1.0.0. L'avanzamento dei processi, dettato dal Piano Di Progetto v1.0.0, é stato mantenuto in controllo tramite una costante verifica delle metriche definite in questo documento e di cui troviamo una rappresentazione grafica in seguito.

5.2 Dettaglio dell'attivitá di verifica su RR

5.2.1 Documenti

Indice di *gulpease* per i documenti redatti:

Documento	**Valore di accettazione**	**Esito**
Piano Di Progetto v1.0.0	>40	*Positivo*
Norme di progetto v1.0.0	>40	*Positivo*
Analisi dei requisiti v1.0.0	>40	*Positivo*
Piano Di Qualifica v1.0.0	>40	*Positivo*
Studio Di Fattibilità v1.0.0	>40	*Positivo*

Tabella 1: esito del calcolo indice di gulpease per ogni documento.

5.2.2 Processi

Sirius ha condotto l'attivitá di verifica per i processi. In questo primo periodo il processo di documentazione é predominante nella pianificazione delle risorse. Di seguito viene riportato l'indice **SV (*schedule variance*)** per le attivitá eseguite e i risultati sono i seguenti:

Attivitá	**Ore pianificate**	**Ore rilevate**	**SV rilevato**	**SV accettazione**
Norme di Progetto	17 H	17 H	0 H	> -1 H
Studio di Fattibilitá	8 H	14 H	**-6 H**	**> -1 H**
Analisi dei Requisiti	70 H	68 H	2 H	> -4 H
Piano di Progetto	37 H	35 H	2 H	> -2 H
Piano di Qualifica	26 H	22 H	4 H	> -2 H

Tabella 2 : Indice SV per le attivitá.

Da una prima analisi, si denota che *Sirius* ha pianificato in modo preciso le attivitá. L'attivita' di Studio di Fattibilitá, essendo stato uno dei primi documenti che *Sirius* ha redatto, la pianificazione non é stata precisa questo ha portato ad un SV dell'attivitá fuori dal range di accettazione. Le cause di questo problema sono da ricercare anche nella poca confidenza con gli strumenti di *editor* testi e con gli strumenti di condivisione. La singola occorrenza del problema, non é quindi indice di allarme per gli altri processi che saranno pianificati nell'avanzamento del prodotto.

- SV-totale = 2 H;

SV-totale maggiore di zero denota che *Sirius* stá producendo piú velocemente rispetto a quanto pianificato. Questo puó essere una diretta conseguenza dell'aggiunta di uno *slack* temporale nella pianificazione delle attivitá. Il team ha valutato la possibilitá di ridurre il tempo di *slack*, per fare in modo che la pianificazione corrisponda alla realtá; ma data la variabilitá delle attivitá che *Sirius* intende svolgere nel proseguo del progetto e la poca esperienza, é stato deciso di non modificare tale valore.
Di seguito viene riportato l'indice **BV (*budget variance*)** per le attivitá eseguite e i risultati sono i seguenti:

Attivitá	**Costo pianificato**	**Costo consuntivo**	**BV rilevato**	**BV limite**
Norme di Progetto	325.00 €	325.00 €	0.00 €	> -32.50 €
Studio di Fattibilitá	180.00 €	310.00 €	**-130.00 €**	> **-18.00 €**
Analisi dei Requisiti	1630.00 €	1600.00 €	30.00 €	> -16.30 €
Piano di Progetto	1005.00 €	945.00 €	60.00 €	> -10.05 €
Piano di Qualifica	490.00 €	420.00 €	70.00 €	> -49.00 €

Tabella 3: Indice BV per le attivitá.

Come descritto sopra per SV, anche BV denota che il preventivo di costo previsto per le attivitá svolte é stato corretto. In particolare nell'attivitá di Studio di Fattibilitá il costo a consuntivo é stato maggiore rispetto a quello preventivato. Questo é da collegare al costo orario dell'amministratore e non meno alle cause elencate sopra per l'indice SV che ne é strettamente collegato. Complessivamente *Sirius* ha ottenuto:

- BV-totale = 30.00 €.

il risultato ottenuto é una diretta conseguenza di un preventivo appropriato, e quindi ad un piccolo margine di guadagno nel budget di spesa dell'intero progetto.

A Appendice

A.1 Ciclo di Deming

Alla luce delle informazioni sopra citate il team ha deciso di adottare la politica del ciclo PDCA per le attivitá da svolgere. Lo stesso, oltre a fornire supporto nella pianificazione garantisce un elevato standard qualitativo tramite il *Miglioramento continuo*, che è alla base del ciclo di Deming.

Figura 1: Ciclo di Deming

- ***Plan***: pianificazione che prevede la definizione di procedure, risorse, scadenze e responsabilitá ;
- ***Do***: esecuzione delle attivitá pianificate;
- ***Check***: controllo dei risultati ottenuti e confronto con quelli pianificati;
- ***Act***: Analisi dei risultati ottenuti e modifica o definizione di nuove procedure che permettano di evitare gli aspetti critici dei processi in esame.

L'adozione del PDCA garantisce un continuo arricchimento dei processi tramite dei cambiamenti e delle riorganizzazioni. Alla base di questo, ci deve essere una conoscenza specifica delle Norme di progetto v1.0.0 da parte di tutti i componenti del team. Inoltre, queste migliorie aumentano i costi di gestione e per questo devono essere valutati dal *Responsabile di progetto*.

A.2 ISO/IEC 9126

Lo standard ISO/IEC 9126 descrive gli obiettivi qualitativi di prodotto e delinea in generale le metriche per misurare il raggiungimento di tale obiettivo (figura 3). In questo standard i criteri sono divisi in 3 aree diverse:

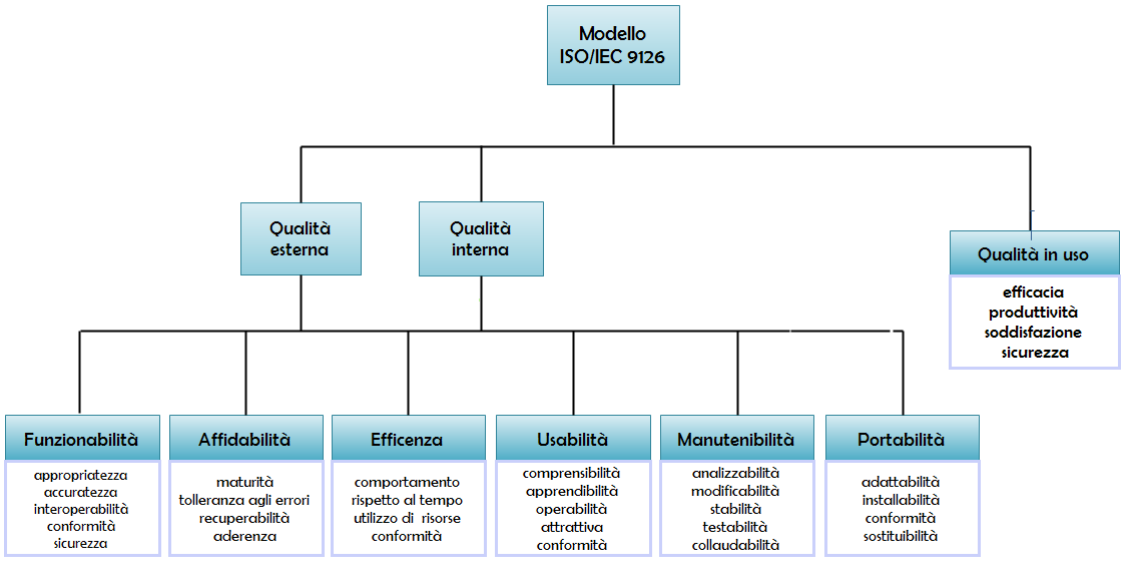

Figura 2: Caratteristiche qualitative definite dal modello ISO/IEC 9126

- **Qualitá in uso**:é la qualitá del software dal punto di vista dell'utilizzatore;
- **Qualitá esterna**:é la qualit'a del software dal punto di vista esterno nel momento in cui esso viene eseguito e testato in ambiente di prova;
- **Qualitá interna**:é la qualitá del software vista dall'interno e quindi sono le caratteristiche implementative del software quali architettura e codice che ne deriva.

Non avendo modo di verificare la qualitá in uso, *Sirius*ha deciso di lavorare su qualitá interna ed esterna definendo apposite metriche.

- **Funzionalitá** é la capacità di un prodotto software di fornire funzioni che soddisfano esigenze stabilite, necessarie per operare sotto condizioni specifiche.
  - Appropriatezza: rappresenta la capacità del prodotto software di fornire un appropriato insieme di funzioni per gli specificati compiti ed obiettivi prefissati all'utente.
  - Accuratezza: la capacità del prodotto software di fornire i risultati concordati o i precisi effetti richiesti;
  - Interoperabilità: è la capacità del prodotto software di interagire ed operare con uno o più sistemi specificati;
  - Conformità: la capacità del prodotto software di aderire a standard, convenzioni e regolamentazioni rilevanti al settore operativo a cui vengono applicate;

  - Sicurezza: la capacitá del prodotto software di proteggere informazioni e dati negando in ogni modo che persone o sistemi non autorizzati possano accedervi o modificarli, e che a persone o sistemi effettivamente autorizzati non sia negato l'accesso ad essi.

- **Affidabilitá**: é la capacità del prodotto software di mantenere uno specificato livello di prestazioni quando usato in date condizioni per un dato periodo.
  - Maturità: é la capacità di un prodotto software di evitare che si verificano errori, malfunzionamenti o siano prodotti risultati non corretti;
  - Tolleranza agli errori: é la capacità di mantenere livelli predeterminati di prestazioni anche in presenza di malfunzionamenti o usi scorretti del prodotto;
  - Recuperabilità: é la capacità di un prodotto di ripristinare il livello appropriato di prestazioni e di recupero delle informazioni rilevanti, in seguito a un malfunzionamento. A seguito di un errore, il software può risultare non accessibile per un determinato periodo di tempo, questo arco di tempo é valutato proprio dalla caratteristica di recuperabilità;
  - Aderenza: è la capacità di aderire a standard, regole e convenzioni inerenti all'affidabilità.

- **Usabilitá**: é la capacità del prodotto software di essere capito, appreso, usato e benaccetto dall'utente, quando usato sotto condizioni specificate. ottimale"
  - Comprensibilità: esprime la facilità di comprensione dei concetti del prodotto, mettendo in grado l'utente di comprendere se il software è appropriato.
  - Apprendibilità: è la capacità di ridurre l' impegno richiesto agli utenti per imparare ad usare la sua applicazione;
  - Operabilità: è la capacità di mettere in condizione gli utenti di farne uso per i propri scopi e controllarne l'uso;
  - Attrattiva: è la capacità del software di essere piacevole per l'utente che ne fa uso;
  - Conformità: è la capacità del software di aderire a standard o convenzioni relativi all'usabilità.

- **Efficienza**: é la capacità di fornire appropriate prestazioni relativamente alla quantità di risorse usate.
  - Comportamento rispetto al tempo: è la capacità di fornire adeguati tempi di risposta, elaborazione e velocità di attraversamento, sotto condizioni determinate;

  - Utilizzo delle risorse: è la capacità di utilizzo di quantità e tipo di risorse in maniera adeguata.
  - Conformità: è la capacità di aderire a standard e specifiche sull'efficienza$_G$.

- **Manutenibilitá**: é la capacità del software di essere modificato, includendo correzioni, miglioramenti o adattamenti.
  - Analizzabilità: rappresenta la facilità con la quale è possibile analizzare il codice per localizzare un errore nello stesso;
  - Modificabilità: la capacità del prodotto software di permettere l'implementazione di una specificata modifica (sostituzioni componenti);
  - Stabilità: la capacità del software di evitare effetti inaspettati derivanti da modifiche errate;
  - Testabilità: la capacità di essere facilmente testato per validare le modifiche apportate al software.

- **Portabilitá**: é la capacità del software di essere trasportato da un ambiente di lavoro ad un altro.
  - Adattabilità: la capacità del software di essere adattato per differenti ambienti operativi senza dover applicare modifiche diverse da quelle fornite per il software considerato;
  - Installabilitá: la capacità del software di essere installato in uno specificato ambiente;
  - Conformitá: la capacità del prodotto software di aderire a standard e convenzioni relative alla portabilità;
  - Sostituibilitá: è la capacità di essere utilizzato al posto di un altro software per svolgere gli stessi compiti nello stesso.

A.3 Capability Maturity Model Integration (CMMI)

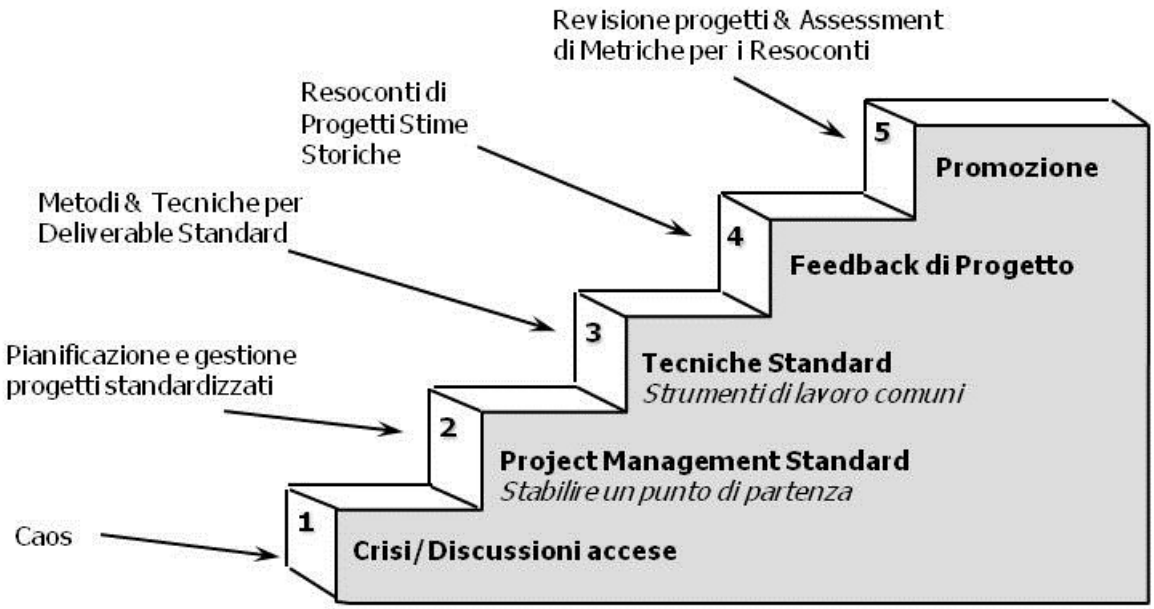

Figura 3: Livello di maturitá delle procedure

Il modello identifica cinque livelli di maturità dei processi all'interno di un'organizzazione: dal Livello 1, il processo$_G$ più immaturo, o caotico, al Livello 5, il processo$_G$ più maturo, o di qualità

- **Livello di maturità 1**
  Partendo dall'assunzione che una pratica non può essere migliorata se non è ripetibile, il livello di maturità iniziale vede l'organizzazione effettuare la gestione delle persone tramite procedure ad hoc, spesso informali e non ripetibili se non sporadicamente. Un esempio tipico è data dall'impossibilità, da parte delle persone, di assicurare la data di rilascio del software, indipendentemente dalle tecnologie utilizzate o dalla preparazione delle persone. Un'altra conseguenza tipica è la gestione incontrollata delle modifiche ai requisiti con conseguenze negative sui piani di lavoro. L'attività principale da compiere in questa fase è quella di aiutare l'organizzazione a rimuovere ogni impedimento alla ripetibilità delle pratiche;

- **Livello di maturità 2**
  Al livello di maturità 2, l'organizzazione stabilisce una politica per divulgare presso tutti i gruppi di lavoro i processi stabiliti. Prima di pensare ad ogni miglioramento, l'organizzazione deve assicurare un ambiente di lavoro stabile in cui eseguire in maniera ripetibile i propri processi. Finché si opera in una modalità non strutturata, il management è troppo occupato nel controllo quotidiano delle

operazione per poter pensare a qualsivoglia cambiamento in ottica di miglioramento. L'obiettivo principale del livello 2 è quindi quello di permettere alle persone di svolgere il proprio lavoro in maniera ripetibile, in base a quanto già fatto in passato ed in base all'esperienza maturata. A questo livello il management lascia ai responsabili dei singoli gruppi il compito di controllare il lavoro quotidiano, dedicandosi a sua volta al controllo dei risultati finali e della baseline (ed alle rispettive modifiche). Solo quando le pratiche stabilite saranno eseguire con naturalezza dall'intera organizzazione, questa potrà iniziare la fase successiva di utilizzo di processi comuni a tutta l'organizzazione;

- **Livello di maturità 3**
  Al livello di maturità 3, l'organizzazione seleziona le migliori pratiche e le include in un processo comune. Operando tutti con le stesse pratiche definite, l'organizzazione sarà in grado di valutare le pratiche con migliori performance nell'ambiente comune. Documentate nell'ambito del processo comune le pratiche, queste diventano anche lo strumento di apprendimento per le nuove persone. Le misure effettuate sulle pratiche di maggiore criticità sono registrate in un archivio ed utilizzare per effettuarne l'analisi. In tale modo si è creato il fondamento per una cultura di base comune all'organizzazione: un processo comune conosciuto ed applicato da tutti. E' il fondamento della cultura professionale di base dell'organizzazione;

- **Livello di maturità 4**
  Al livello di maturità 4, l'organizzazione inizia a gestire i processi in base ai risultati utilizzando l'analisi delle misure effettuate. Le attività sono svolte secondo i processi comuni definiti ed i risultati sono quindi più controllabili in base all'esperienza storica. Le deviazioni dai risultati attesi sono analizzate, le cause delle deviazioni individuate e le azioni correttive prese di conseguenza. I processi sono quindi gestiti quantitativamente ed i risultati sono prevedibili con maggiore cura. I risultati del business sono controllati da valori e non più dalle $milestone_G$ come prima. Si crea quindi la cultura per un vero miglioramento dei processi e quindi delle performance reali;

- **Livello di maturità 5**
  Al livello di maturità 5, l'organizzazione opera utilizzando in maniera ripetitiva i propri processi, ne valuta le performance quantitativamente ed opera per migliorarli di continuo. Gli eventuali difetti sono analizzati e le cause che li generano sono rimosse per evitare il loro ripetersi. Le persone sono culturalmente abituate ad eseguire i processi conosciuti ed il management a gestirli quantitativamente ed a migliorarli. Si crea anche la cultura dell'accettazione del cambiamento. L'organizzazione entra in un circolo virtuoso di miglioramento continuo;