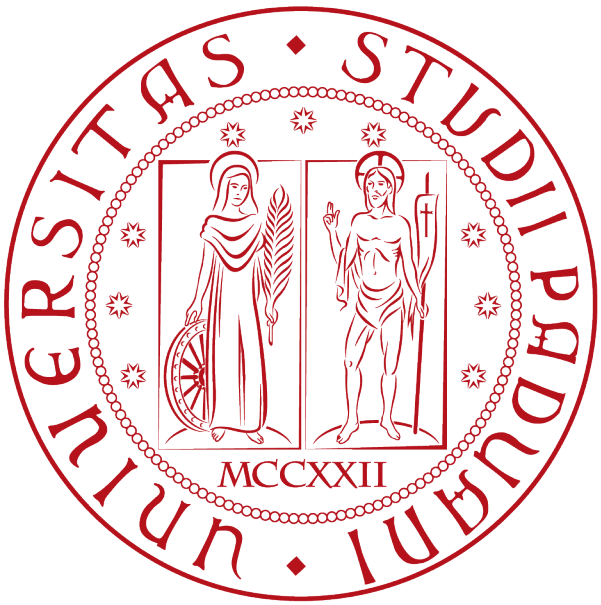

Piano di Qualifica

SonsOfSwe - Progetto Marvin

sonsofswe.swe@gmail.com

Informazioni sul documento

Versione	1.0
Redazione	Caldart Federico Favero Andrea Menegon Lorenzo
Verifica	- -
Approvazione	- -
Uso	interno
Distribuzione	Vardanega Tullio Cardin Riccardo Gruppo SonsOfSwe

Descrizione

Questo documento descrive le regole, gli strumenti e le convenzioni adottate dal gruppo SonsOfSwe durante la realizzazione del progetto Marvin.

Indice

1 Introduzione

1.1 Scopo del documento

Questo documento ha lo scopo di descrivere gli obiettivi di qualità, di processo e di prodotto da raggiungere nella realizzazione del progetto, e le strategie di verifica e validazione adottate per il raggiungimento di tali obiettivi.

1.2 Scopo del prodotto

L'obiettivo di Marvin è di realizzare un *prototipo*$_G$ di Uniweb come *ĐApp*$_G$ che giri su *Ethereum*$_G$. I tre attori principali che si rapportano con Marvin sono:

1. Università;
2. Professori;
3. Studenti.

Il portale deve quindi permettere agli studenti di accedere alle informazioni riguardanti le loro carriere universitarie, di iscriversi agli esami, di accettare o rifiutare voti e di poter vedere il loro libretto universitario. Ai professori deve invece essere permesso registrare i voti degli studenti. L'università ogni anno crea una serie di corsi di laurea rivolti a studenti, dove ognuno di essi comprende un elenco di esami disponibili per anno accademico. Ogni esame ha un argomento, un numero di crediti e un professore associato. Gli studenti si iscrivono ad un corso di laurea e tramite il libretto elettronico mantengono traccia ufficiale del progresso.

1.3 Glossario

Nel documento Glossario i termini tecnici, gli acronimi e le abbreviazioni sono definiti in modo chiaro e conciso, in modo tale da evitare ambiguità e massimizzare la comprensione dei documenti.
I vocaboli presenti in esso saranno posti in corsivo e presenteranno una "G" maiuscola a pedice.

1.4 Riferimenti

1.4.1 Normativi

- **Capitolato d'appalto C6 - Marvin: dimostratore di Uniweb su Ethereum** http://www.math.unipd.it/~tullio/IS-1/2017/Progetto/C6.pdf;
- **NormeDiProgetto v0.0.1**.

1.4.2 Informativi

- **Piano di progetto:** Piano di progetto;
- **Slides del corso di Ingegneria del software** http://www.math.unipd.it/~tullio/IS-1/2017/
- **Ingegneria del software, decima edizione - Ian Sommerville, capitolo 21**
- **SLOC** https://it.wikipedia.org/wiki/Source_lines_of_code
- **Ciclo di Deming** https://it.wikipedia.org/wiki/Ciclo_di_Deming (aggiornato al 31/03/2018)
- **Indice Gulpease** https://it.wikipedia.org/wiki/Indice_Gulpease (aggiornato al 31/03/2018)
- **Complessità ciclomatica** https://it.wikipedia.org/wiki/Complessità_ciclomatica (aggiornato al 31/03/2018)

2 Obiettivi di qualità

Questa sezione ha l'obiettivo di definire le caratteristiche riguardanti la qualità di prodotto e di processo che dovranno essere perseguite durante lo sviluppo del progetto. Ogni caratteristica viene valutata da una metrica, una soglia di accettabilità, ed una possibile soglia di miglioramento che il *team*$_{\text{G}}$ si prefigge di raggiungere e possibilmente superare.

2.1 Qualità di processo

La qualità di processo influenza direttamente il prodotto finale realizzato. É necessario quindi sviluppare un processo in grado di produrre ciclicamente un prodotto di alta qualità. Per questo motivo si è deciso di stabilire le seguenti caratteristiche da rispettare per tutto lo sviluppo del progetto, contemporaneamente a:

- L'applicazione del *Ciclo di Deming*$_{\text{G}}$, o *PDCA*, al fine di perseguire il miglioramente continuo delle attività di processo.
- L'adesione allo standard ISO/IEC 15504, denominato *SPICE*$_{\text{G}}$, al fine di applicare una valutazione oggettiva sulla maturità dei processi.

2.1.1 Pianificazione

La pianificazione temporale necessita di uno sguardo a ritroso a partire dagli obiettivi prefissati per completare in tempo adeguato il lavoro previsto. Per un *team*$_{\text{G}}$ è fondamentale rispettare le scadenze previste, e nel caso in cui si verifichi una situazione di possibile ritardo si rischia di violare l'obiettivo di qualità prefissato, e andranno effettuati quindi dovuti controlli.

- **Metrica**: Si è deciso di utilizzare la **Schedule Variance**$_{\text{G}}$. (Sarebbe da aggiungere il link direttamente all'appendice per l'SV)
- **Soglia di accettabilità**: Si è deciso di ritenere accettabile un ritardo di massimo 3 giorni lavorativi rispetto a quanto specificato nel "*Piano di Progetto*".
- **Soglia di ottimalità**: Si ritiene un miglioramento rispetto all'obiettivo prefissato il caso in cui un lavoro venga portato a termine almeno 2 giorni lavorativi prima del dovuto, in termini di guadagno di tempo complessivo.

2.1.2 Miglioramento

Al fine di valutare e migliorare la qualità del lavoro svolto è stato assunto il modello di riferimentoper la valutazione del livello di maturità definito da SPICE.

- **Metrica**: Verrà utilizzata la struttura a 6 livelli che rappresenta la scala di maturità; la misura di ogni livello sarà effettuata con i 4 livelli N,P,L,F definiti dallo standard.
- **Soglia di accettabilità**: Il livello minimo accettabile di maturità della scala in riferimento ai processi è il 2 (Managed); il processo deve cioè fornire i risultati conformi agli standard ed ai requisiti iniziali in maniera pianificata e tracciabile.
- **Soglia di ottimalità**: La soglia di ottimalità verrà raggiunta con il livello 4 (Predictable); il processo dovrà cioè essere eseguito in conformità ai principi dell'ingegneria del software e attuato all'interno di limiti ben definiti.

2.1.3 Costo

Per verificare se i costi sono stati rispettati con quanto concordato nel *"Piano di Progetto"*, è stato deciso di utilizzare la ***Cost Variance***$_{\text{G}}$ (CV). Qualora un processo non possieda la qualità minima concordata,

necessiterà di lavoro aggiuntivo al fine di soddisfare i requisiti richiesti ma alzando il costo complessivo del progetto, che sarà valutato secondo i seguenti parametri:

- **Metrica**: L'unità di misura scelta per valutare l'aumento dei costi stabiliti è la Cost Variance.
- **Soglia di accettabilità**: Sarà accettabile un aumento dei costi superiore a quelli previsti nel *"Piano di Progetto"* di un massimo del 10%
- **Soglia di ottimalità**: La soglia di ottimalità verrà raggiunta nel caso in cui i costi non aumenteranno rispetto a quanto concordato nel *"Piano di Progetto"*,

Per informazioni più approfondite riguardo lo standard ISO/IEC 15504, si rimanda alla sezione Standard di processo: ISO/IEC 15504 - Software Process Improvement and Capability Determination dell'appendice A.

2.2 Qualità di prodotto

2.2.1 Qualità di documento

Il team si impegna a redigere dei documenti di alta qualità, rispettando le carratestiche di forma e contenuto descritte di seguito.

2.2.1.1 Ortografia Un documento poichè possa essere definito tale, deve essere prima di tutto privo di errori dal punto di vista grammaticale e ortografico. Il primo controllo avverà proprio durante la stesura del documento stesso, tramite il sistema di autocontrollo dell'ambiente *"TexStudio"*, per poi essere controllato dal *Verificatore*$_{\mathrm{G}}$.

- **Metrica**: L'unità di misura considerata è il numero di errori ortografici riscontrati dopo il primo controllo da parte del *Verificatore*.
- **Soglia di accettabilità**: Si è accettata come tollerabile la presenza di massimo un 5% di errori rispetto alla quantità totale segnalata dopo la prima analisi da parte del *Verificatore*.
- **Sogia di ottimalità**: La soglia di ottimalità verrà raggiunta nel caso in cui dopo la prima revisione del documento non vengano più riscontrati errori dal *Verificatore* e dal *Responsabile*.

L'argomento verrà trattato dettagliatamente nella sezione Errori ortografici (ci va il link) in appendice.

2.2.1.2 Comprensibilità e leggibilità Poichè un documento venga considerato leggibile e scorrevole si è deciso di adottare l'*Indice Gulpease*$_{\mathrm{G}}$, al fine di avere un parametro oggettivo e facilmente misurabile.

- **Metrica**: L'unità di misura utilizzata è l'*Indice Gulpease*.
- **Soglia di accettabilità**: Verrà considerato come accettabile un valore di 45 sulla scala dell'*Indice Gulpease*.
- **Soglia di ottimalità**: La soglia di ottimalità verrà raggiunta nel caso in cui l'*Indice Gulpease* sia maggiore di 60.

L'argomento verrà trattato dettagliatamente nella sezione Indice Gulpease (ci va il link) in appendice.

2.2.1.3 Correttezza dei contenuti Oltre che ad essere corretto nella forma, un documento necessita di un contenuto adeguato dal punto di vista argomentativo. Gli *Analisti* saranno direttamente responsabili della qualità del contenuto, che poi verrà controllato e corretto dal *Verificatore*. Per verificare la correttezza concettuale dei documenti prenderemo in esame i seguenti parametri:

- **Metrica**: La quantità di errori presente dopo la prima verifica del documento sarà l'unità di misura presa in considerazione.
- **Soglia di accettabilità**: La soglia di accettabilità è fissata ad una quantità di errori pari al 5% rispetto alla precedente verifica del documento.

- **Soglia di ottimalità**: La soglia di ottimalità sarà raggiunta nel caso in cui non si riscontrino errori dopo la prima verifica del documento.

2.2.1.4 Adesione alle norme interne Al fine di ottenere un prodotto coerente ogni documento dovrà essere redatto rispettando strettamente quanto dichiarato nelle *Norme di Progetto*. Qualunque riferimento non attinente o in contrasto a quanto dichiarato verrà considerato un errore.

- **Metrica**: La quantità di errori presente dopo la prima verifica del documento sarà l'unità di misura presa in considerazione.
- **Soglia di accettabilità**: La soglia di accettabilità massima è fissata ad una quantità di errori pari al 5% rispetto alla precedente verifica del documento.
- **Soglia di ottimalità**: La soglia di ottimalità sarà raggiunta nel caso in cui non si riscontrino errori dopo la prima verifica del documento.

Per una precisa definizione degli errori in riferimento alle norme interne si veda la sezione Errori di forma (ci va il link) in appendice.

2.2.2 Qualità del software

Come detto in precedenza, è impossibile distinguere in maniera netta la qualità di processo dalla qualità del software, in quanto la prima influenza direttamente la seconda; è dunque fondamentale avere alla base una qualità di processo sufficientemente buona per garantire la qualità del prodotto. Nonostante ciò, è necessario stabilire degli obiettivi quantitativi di qualità del software oggettivi e misurabili. A tal fine verrà seguito lo standard ISO/IEC 9126, il quale si sostanzia nei sei punti seguenti:

2.2.2.1 Funzionalità È un requisito funzionale che indica la capacità del software di soddisfare le esigenze esposte dal capitolato ed individuate durante l' *Analisi dei Requisiti*. Per valutare la funzionalità del software prenderemo in considerazione i seguenti parametri:

- **Metrica**: La valutazione si baserà sul numero di requisiti soddisfatti.
- **Soglia di accettabilità**: Il prodotto verrà valutato come accettabile se tutti i requisiti obbligatori saranno soddisfatti.
- **Soglia di ottimalità**: La soglia di ottimalità sarà raggiunta nel caso in cui siano soddisfatti sia i requisiti obbligatori che tutti i requisiti opzionali.

2.2.2.2 Affidabilità È un requisito non funzionale che indica la capacità del software di svolgere correttamente il suo compito, mantenendo delle buone prestazioni anche al variare dell'ambiente nel tempo. Per valutare l'affidabilità del software prenderemo in considerazione i seguenti parametri:

- **Metrica**: La valutazione si baserà sul numero di fallimenti durante la fase si test.
- **Soglia di accettabilità**: Il prodotto verrà valutato come accettabile se i test falliti saranno inferiori o uguali al 5%.
- **Soglia di ottimalità**: La soglia di ottimalità sarà raggiunta nel caso in cui il 100% dei test darà l'esito desiderato.

2.2.2.3 Efficienza Oltre al tempo anche costo di ether??? È un requisito non funzionale che valuta la capacità di un prodotto software di realizzare le funzioni richieste nel minor tempo possibile e con l'uso minimo di risorse necessarie.

- **Metrica**: La valutazione si baserà sui secondi impiegati dal prodotto per eseguire le richieste dell'utente.
- **Soglia di accettabilità**: La soglia di accettabilità è il periodo tra 0 e 10 secondi.

- **Soglia di ottimalità**: La soglia di ottimalità è 1 secondo.

2.2.2.4 Usabilità L'usabilità è un requisito non funzionale che indica la capacità del software di essere capito e usato correttamente da parte dell'utente finale. Dato che il prodotto finale sarà per l'utente un portale web, è impossibile trovare una metrica quantificabile per valutarne l'usabilità: essa dipende da molteplici fattori che coinvolgono anche le capacità dell'utente stesso e gli strumenti a sua disposizione. Verrà dunque valutata in modo oggettivo basandosi sugli standard del web dichiarati dal *W3C*$_{\text{G}}$ e sugli strumenti che tale organizzazione mette a disposizione, al fine di creare un'interfaccia web il più accessibile possibile. Prenderemo in considerazione i seguenti parametri:

- **Metrica**: La valutazione si baserà sul numero di errori trovati dagli strumenti del W3C.
- **Soglia di accettabilità**: La soglia di accettabilità è di 2 errori rilevati.
- **Soglia di ottimalità**: Il prodotto sarà dichiarato ottimo se saranno rilevati 0 errori.

Si rimanda alla sezione dell'appendice per maggiori informazioni sulla metrica utilizzata.

Quanto detto non assicura però una valutazione completa dell'usabilità, la quale è soggettiva; sarà necessario dunque predisporre test specifici per la misurazione, coinvolgendo ad esempio persone esterne al gruppo al fine di stabilire quanto mediamente il software sia capibile. Al momento il team non è tuttavia in grado di stabilire con precisione una metrica adatta a misurare questo risultato.

2.2.2.5 Manutenibilità La manutenibilità è un requisito non funzionale che indica la capacità di un prodotto di essere evolvibile nel tempo attraverso correzioni, miglioramenti e aggiunte.

- **Metrica**: Saranno usate le metriche riguardanti il codice, dato che esso influenza direttamente la manutenibilità del software.
- **Soglia di accettabilità**: La soglia di accettabilità sarà raggiunta se il prodotto raggiungerà tale soglia in tutte le metriche utilizzate per il codice.
- **Soglia di ottimalità**: La soglia di ottimalità sarà raggiunta se il prodotto raggiungerà tale soglia in tutte le metriche utilizzate per il codice.

Si rimanda alla sezione dell'appendice per maggiori informazioni.

2.2.2.6 Portabilità La portabilità è un requisito non funzionale che indica la capacità del prodotto di operare in *ambienti*$_{\text{G}}$ diversi, limitando le necessità di apportare cambiamenti.

- Metrica: La valutazione si baserà sul numero di versioni di *browser*$_{\text{G}}$ e numero di browser stessi su cui il prodotto riesce a venire utilizzato e visualizzato correttamente.
- Soglia di accettabilità: La soglia di accettabilità sarà raggiunta se il prodotto sarà supportato correttamente, offrendo la totalità delle sue funzionalità, dalla versione aggiornata dei browser *Google Chrome*$_{\text{G}}$, *Microsoft Edge*$_{\text{G}}$, *Mozilla Firefox*$_{\text{G}}$, *Safari*$_{\text{G}}$ e *Opera*$_{\text{G}}$ su *dekstop*$_{\text{G}}$.
- Soglia di ottimalità: La soglia di ottimalità sarà raggiunta se il prodotto sarà supportato correttamente, offrendo la totalità delle sue funzionalità, in aggiunta ai sopra citati, da *Internet Explorer 11*$_{\text{G}}$ su desktop e da Google Chrome e Safari nelle versioni *mobile*$_{\text{G}}$ aggiornate.

Per informazioni più approfondite riguardo lo standard ISO/IEC 9126, si rimanda alla sezione Standard di prodotto: SO/IEC 9126 dell'appendice A.

A Qualità secondo gli standard

Al fine di perseguire la qualità secondo quanto descritto in questo documento, ci si basa alcuni standard, in modo da bilanciare la poca esperienza del team con la conoscenza ricavata da anni di pratica nell'ambito dell'ingegneria del software, trascritta in tali documenti.

A.1 Standard di processo: ISO/IEC 15504 - Software Process Improvement and Capability Determination

La qualità di un prodotto software (ovvero le caratteristiche di un prodotto che gli permettono di soddisfare delle esigenze (implicite od esplicite))dipende dalla qualità dei suoi processi. L'*ISO/IEC 15504 - Software Process Improvement and Capability Determination* o *SPICE* è uno standard che permette di valutare i processi software di un prodotto con lo scopo di migliorarli (in modo continuativo). La valutazione dei processi permette di identificare in modo indipendente la *capacità*$_G$ (capability) di ciascuno di essi attraverso i loro attributi, gli esiti di ogni valutazione. Basandoci su tali risultati di valutazione (che devono essere comparabili, ripetibili ed oggettivi)ci si può aspettare un miglioramento *continuativo* dei processi identificandone i punti di forza, di debolezza ed anche i rischi ed i modi per prevenire questi ultimi.

glossarizzare

Ad ogni processo viene assegnato un livello di capacità a seconda della classificazione dei suoi attributi

> **0 - Incomplete**: il processo presenta una incapacità generale nel raggiungere il proprio obbiettivo. A questo livello di capacità non viene associato alcun attributo.
>
> **1 - performed**: il processo è riuscito a raggiungere il proprio obbiettivo. Il raggiungimento di tale obbiettivo potrebbe non essere stato pianificato e tracciato in modo rigoroso. A questo livello è associto l'attributo **process performance**.
>
> **2 - managed**: il processo (che appartiene anche al livello 1) rilascia i propri prodotti secondo procedure specifiche ed è pianificato e tracciato. I prodotti sono conformi agli standard specificati ed ai requisiti. A questo livello sono associati due attributi: **performance management** e **work product management**.
>
> **3 - established**: il processo (che appartiene anche al livello 2) viene implementato utilizzando dei buoni principi di ingegneria del sofware ed è in grado di raggiungere ogni volta che viene eseguito i medesimi risultati. A questo livello sono associati due attributi: **process definition** e **process deployment**.
>
> **4 - predictable**: il processo (che appartiene anche al livello 3) viene eseguito nella pratica in modo coerente rimanendo dentro ai limiti di controllo che, sono stati definiti per raggiungere il suo obbiettivo. Il livello ha associati gli attributi **process controll** e **process measurement**
>
> **5 - optimizing**: le performance del processo (che appartiene anche al livello 4) sono ottimizzate in modo continuo per andare incontro agli obbiettivi ed alle necessità (bisogni) di progetto o di business aziendali presenti e futuri. Anche a questo livello sono associati due attributi: **process innovation** e **process optimization**.

I 9 attributi che servono per misurare la capacità di un processo sono definiti nel seguente modo:

> **Process performance**: è una misura che indica il raggiungimento degli obbiettivi del processo.
>
> **Performance management**: è una misura che indica come sono gestite le performance del processo.
>
> **Work product management**: è una misura che indica quanto i prodotti del processo siano gestiti in modo appropriato.
>
> **Process definition**: è una misura che indica quanto il processo sia effettivamente impegnato a rispettare gli standard quando produce i propri esiti.
>
> **Process deployment**: è una misura di quanto il processo standard venga diffuso efficacemente per raggiungere i popri risultati.

Process measurement: è una misura che indica quanto vengono usate le misurazioni dei risultati del processo per assicurarsi che le sue performance supportino il raggiungimento degli obbiettivi aziendali fissati.

Process control: è una misura che dà una indicazione di quanto il processo sia gestito in modo quantitativo, per produrre un processo che sia stabile, capace[1] e prevedibile entro i limiti definiti.

Process innovation: è una misura di quanto i cambiamenti al processo sono identificati grazie ad analisi di cause comuni delle variazioni delle performance e da indagini di approcci innovativi per le definizioni e lo sviluppo dei processi.

Process optimization: è una misura che indica quanto i cambiamenti alla definizione, gestione e performance del processo abbiano un impatto effettivo che permetta di raggiungere gli obiettivi rilevanti di miglioramento del processo.

Ogni attributo di processo viene valutato attraverso una scala di valutazione di quattro livelli[2]. Il punteggio è basato sulle prove raccolte tramite degli indicatori che permettono di sapere quale livello della classifica si attesta l'attributo.

N - Not achieved: (0% - 15%)

P - Partially achieved: (>15% - 50%)

L - Largely achieved: (>50% - 85%)

F - Fully achieved: (>85% - 100%).

Per raggiungere un certo livello di capacità, tutti gli attributi di processo del livello in questione devono essere realizzati almeno come "L" e tutti gli attributi di tutti i livelli di capacità sottostanti devono essere "F".

In Figura 1 sono rappresentati gli ultimi cinque livelli di capacità dei processi di SPICE ed i relativi attributi ad essi associati.

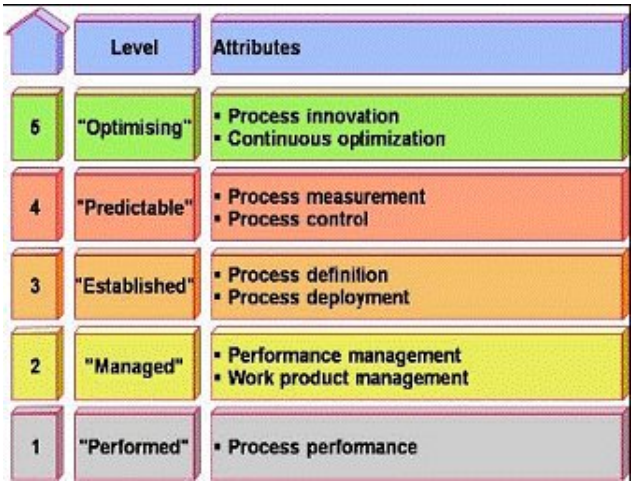

Figura 1: Ultimi 5 livelli di capacità di SPICE

A.2 Standard di prodotto: SO/IEC 9126

Per valutare la qualità del prodotto software si è deciso di utilizzare lo standard ISO/IEC 9126.

É suddiviso in 4 parti:

[1] vedere definizione di *capacità*$_G$

[2] Descritti nella parte 3 dello standard

Quality Model: caratteristiche di qualità che possono essere usate per descrivere i fattori di qualità di un prodotto software

External Metrics: metriche non misurabili direttamente che possono essere usate per valutare se il prodotto software è conforme al modello di qualità

Internal Metrics: metriche direttamente misurabili utilizzabili per valutare le external metrics

Quality in Use Metrics: Metriche rivolte alla valutazione del sottoinsieme di caratteristiche di qualità legate all'utente

Secondo lo standard sono necessari tre punti di vista per valutare la qualità del prodotto software:

- **percepita/in uso**: è correlata a ciò che percepisce l'utente e per questo motivo definisce delle metriche che possono essere applicate solamente quando il prodotto è finito e che esprimono l'efficacia e l'efficienza con cui il software serve le esigenze del suo utilizzatore.
- **esterna**: rappresenta le prestazioni del prodotto e le funzionalità che esso offre. Definisce delle metriche che esprimomo il comportamento dinamico del software, per questo motivo è rilevata attraverso l'analisi dinamica e determina la qualità in uso. É una misura dell'interazione tra il cliente ed il prodotto in un contesto d'uso specifico, permette di osservare il comportamento del software mentre questo viene utilizzato.
- **interna / intrinseca**: rappresenta le qualità intrinseche del prodotto, ovvero quelle misurabili direttamente dal codice sorgente attravero un'analisi di tipo statico. Si realizza partendo dalle specifiche di qualità fornite dall'utente e le specifiche tecniche tradotte dallo sviluppatore nell'architettura del software.

Gli attributi di qualità interni, influenzano alcuni degli attributi di qualità esterni e quest'ultimi influenzano quelli della qualità in uso.

Per descrivere i tre punti di vista di cui sopra, lo standard definisce due modelli, uno che riguarda le qualità interna ed esterna, ed un altro che riguarda la qualità in uso. Tali modelli presentano due livelli (primo e secondo) di caratteristiche che definiscono la qualità. Le caratteristiche del secondo livello sono sottocaratteristiche del primo livello e vengono valutate rispetto a delle metriche interne ed esterne. In Figura 2 e in Figura 3 si possono vedere le caratteristiche associate ai due modelli.

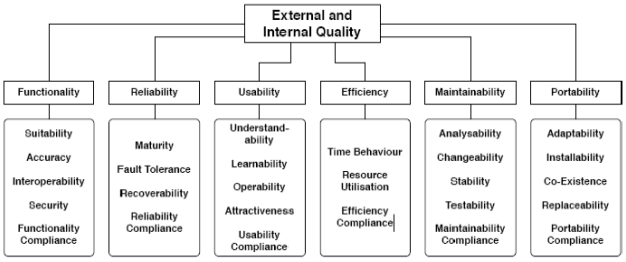

Figura 2: Caratteristiche associate al modello di qualità interna ed esterna

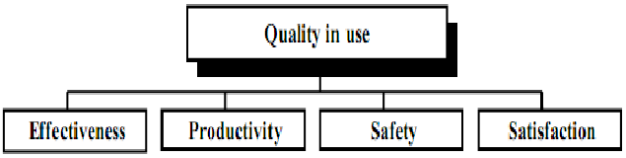

Figura 3: Caratteristiche associate al modello di qualità in uso

Di seguito sono elencate le caratteristiche di primo e secondo livello del modello per la qualità interna ed esterna.

Functionality (funzionalità): La capacità del prodotto deve fornire delle funzioni che soddisfino esigenze stabilite (ovvero emerse dall'Analisi dei requisiti) . mettere ref doc

- **Suitability (appropriatezza)**: capacità del prodotto di fornire all'utente delle funzioni in grado di soddisfare le esigenze stabilite ed implicite.
- **Accuracy (accuratezza)**: capacità del prodotto di fornire risultati corretti con la precisone richiesta.
- **Interoperability (interoperabilità)**: capacità del prodotto di interagire con uno o più sistemi specificati.
- **Security (sicurezza)**: capacità del prodotto di proteggere i dati e le informazioni in modo che persone/sistemi non autorizzate/i riescano ad accedervi in lettura o scrittura-
- **Compliance (conformità)**: capacità del prodotto di aderire agli standard relativi alle funzionalità che offre.

Reliability (affidabilità): Il prodotto software deve mantenere un livello di prestazioni specificato quando viene eseguito sotto certe condizioni specificate

- **Maturity (maturità)**: capacità del prodotto di evitare anomalie.
- **Fault tolerance (tolleranza all'errore)**: capacità del prodotto di mantenere un livello prestazioni specificato nel caso occorrano anomalie.
- **Recoverability (recuperabilità)**: capacità del prodotto di recuperare un livello di prestazioni specificato ed i dati colpiti da dei malfunzionamenti.
- **Compliance (conformità)**: capacità del prodotto di aderire a standard relativi all'affidabilità.

Usability (usabilità): Il prodotto software compreso ed utilizzato con gradimento dall'utente

- **Understandability (comprensibilità)**: il prodotto software permette di capire all'utente se può essergli utile per dei compiti particolari.
- **Learnability (apprendibilità)**: il prodotto è in grado di far apprendere all'utente come utilizzare l'applicazione.
- **Operability (operabilità)**: il prodotto permette all'utente di utilizzarlo e di esercitarne il controllo.
- **Attactiveness (attrattività)**: la capacità del prodotto di attrarre l'utente suscitandone un certo livello di gradimento.
- **Compliance (conformità)**: la capacità del prodotto di aderire agli standard di usabilità.

Efficency (efficienza): Il prodotto sfrutta al massimo le risorse di cui necessita per espletare le proprie funzioni.

- **Time behaviour (comportamento nel corso del tempo)**: capacità del prodotto di fornire un tempo di risposta appropriato quando esegue le proprie funzioni.
- **Resource utilisation (utilizzo delle risorse)**: la capacità del prodotto di usare la giusta quantità ed il tipo di risorse quando il esegue le proprie funzioni.
- **Compliance (conformità)**: capacità del prodotto di soddisfare gli standard relativi all'effecienza.

Maintainability (manutenibilità): capacità del prodotto di evolversi nel tempo grazie a delle modifiche o correzioni.

- **Analysability (analizzabilità)**: la capacità del prodotto di essere analizzato per scovare le cause dei malfunzionamenti.
- **Changeability (modificabilità)**: la capacità del prodotto di essere modificato.
- **Stability (stabilità)**: capacità del software di evitare malfunzionamenti dopo essere stato modificato.

- **Testability) (testabilità)**: capacità del software modificato di essere verificato e validato.
- **Compliance (conformità)**: capacità del prodotto di soddisfare gli standard relativi alla manutenibilità.

Portability (portabilità): Il software deve poter essere trasferito da un ambiente ad un altro con l'avanzare delle nuove tecnologie.

- **Adaptability (adattabilità)**: la capacità del prodotto di adattarsi ad ambienti diversi senza che ci sia il bisogno di applicare azione alcuna o di utilizzare mezzi diversi da quelli che sono stati già forniti.
- **Installability (instabilità)**: la capacita del prodotto di essere installato in un ambiente specifico.
- **Co-existence (coesistenza)**: la capacita del prodotto di coesistere con altri prodotti software indipendenti in uno stesso ambiente condividendone le risorse.
- **Replaceability (sostituibilità)**: la capacita del prodotto di sostituire un altro prodotto software che presenta gli stessi scopi nello stesso ambiente.
- **Compliance (conformità)**: capacità del prodotto di soddisfare gli standard relativi alla portabilità.

Di seguito sono elencate le caratteristiche del modello per la qualità in uso che, rappresentano il punto di vista dell'utente sulla qualità del prodotto software:

- **Effectiveness (efficacia)**: permette agli utenti di raggiungere il proprio obbiettivo portandolo a termine con accuratezza e completezza.
- **Productivity (produttività)**: la capacita del prodotto di utilizzare una adeguata quantità di risorse garantendo efficienza.
- **Satisfaction (soddisfazione)**: la capacita del prodotto software di soddisfare gli utenti.
- **Safety (sicurezza)**: la capacita del prodotto di raggiungere livelli accettabili di rischio di danni a persone, software e ambiente operativo su cui è installato.

A.3 Ciclo di Deming

Il ciclo di Deming, chiamato anche PDCA (Plan, Do, Check and Action.) è uno strumento che permette di realizzare il miglioramento continuo della qualità dei processi e quindi anche dei loro prodotti.

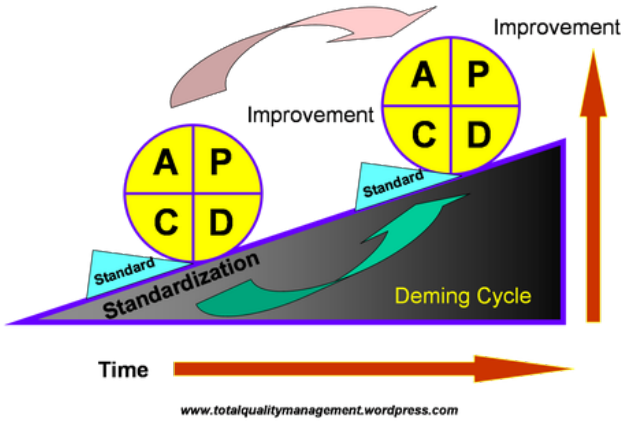

Figura 4: Ciclo di Deming

Come si può vedere in Figura 4 bisogna ripetere in modo iterativo i quattro passi *Plan*, *Do*, *Check* e *Action* per ottenere il miglioramento di un processo.

- **Plan**: vengono pianificati gli obbiettivi per il miglioramento del processo. In questa fase viene analizzata la situazione attuale, vengono raccolti i dati e sviluppate delle metodologie per ottenere dei miglioramenti. Vengono definite le attività che bisogna svolte, le risorse di cui esse necessitano e si fissano le scadenze.

  Per fare ciò è necessario porsi tre domande durante la fase di planning:

  - Cosa si sta cercando di realizzare?
  - Quali cambiamenti si possono fare per ottenere un miglioramento?
  - Come si è in grado di capire che un dato cambiamento rappresenta un miglioramento?
- **Do**: viene attuato il piano definito nella fase di Plan, in questo modo il processo viene eseguito e così viene creato il prodotto.
- **Check**: viene controllato che il processo funzioni come pianificato, in particolare si confrontano i risultato misurati nella fase di Do con gli obbiettivi stabiliti nella fase di Plane (ovvero i risultati attesi).
- **Action**: il processo viene migliorato attuando se necessario delle azioni correttive che devono agire sulle differenze riscontrate tra i risultati attesi e quelli misurati.

Quando tutte queste quattro fasi vengono portate a termine con il massimo della soddisfazione, il miglioramento viene standardizzato. Il prodotto standardizzato è il risultato dell'iniziativa di miglioramento. É possibile che con il cambiamento di alcune circostanze, il processo sia soggeto ad un nuovo miglioramento, in questo modo il ciclo di deming viene ripetuto di volta in volta.

B Metriche

B.1 Metriche per il processo

In questa sezione verranno descritte le metriche che verranno utilizzate per garantire la qualità dei processi.

B.1.0.1 Schedule Variance - SV

La Schedule Variance è un indicatore permette di capire se un processo è il linea con la scheduazione temporale indicata del *Piano di Progetto* o in anticipo o ritardo. Viene calcolata come differenza tra BCWP (Budgeted Cost of Work Performed), cioè il valore, delle attività svolte fino al momento del calcolo e BCWS (Budgeted Cost of Work Scheduled), cioè il valore delle attività che dovrebbe essere state completate secondo la schedulazione preventivata:

$$\mathrm{SV} = \mathrm{BCWP} - \mathrm{BCWS}$$

I risultati possibili sono tre:

- $\mathrm{SV} > 0$, che indica un anticipo sui tempi pianificati.
- $\mathrm{SV} = 0$, che indica l'essere in linea con i tempi pianificati.
- $\mathrm{SV} < 0$, che indica un ritardo sui tempi pianificati.

B.1.0.1.1 Soglie

- Accettabilità: sarà accettato un valore $\mathrm{SV} \geq 0$.
- Ottimalità: sarà ottimo un valore $\mathrm{SV} > 0$.

B.1.0.2 Cost Variance - CV

La Cost Variance è una metrica che permette di capire se i costi effettivi sono in linea o meno con i costi pianificati nel *Piano di Progetto*. Viene calcolata come differanza tra BCWP (Budgeted Cost of Work Performed), cioè il valore delle attività svolte fino al momento del calcolo e ACWP (Actual Cost of Work Performed), cioè il costo effettivamente sostenuto.

$$\mathrm{SV} = \mathrm{BCWP} - \mathrm{ACWP}$$

I risultati possibili sono tre:

- $\mathrm{CV} > 0$, che indica che il progetto sta producendo con un minor costo rispetto a quanto pianificato.
- $\mathrm{CV} = 0$, che indica l'essere in linea con i costi preventivati.
- $\mathrm{CV} < 0$, che indica che il progetto sta producendo con un costo maggiore rispetto a quello pianificato.

B.1.0.2.1 Soglie

- Accettabilità: sarà accettato un valore $\mathrm{CV} \geq 0$.
- Ottimalità: sarà ottimo un valore $\mathrm{CV} > 0$.

B.2 Metriche per il prodotto

B.2.1 Metriche per i documenti

In questa sezione vengono descritte le metriche che verranno utilizzate nel processo di verifica dei documenti prodotti.

B.2.1.1 Errori ortografici

Questa è la metrica che serve ad esprimere un giudizio di correttezza ortografica riguardo il documento prodotto. Gli errori saranno individuati secondo le seguenti modalità: il primo controllo avverrà a tempo di stesura del documento tramite lo strumento di autocorrezione dell'ambiente *"TexStudio"*, mentre il secondo controllo avverrà dopo aver terminato la prima redazione del documento stesso, tramite una verifica manuale del *Verificatore*$_G$. Questa metrica misura il numero di errori riscontrati, attraverso le due modalit'a di verifica, ma non corretti immediatamente.
Formula:

$$\text{Errori ortografici} = \frac{\text{numero errori non corretti}}{\text{numero totale errori segnalati}} * 100$$

B.2.1.1.1 Soglie

- Accettabilità: valore inferiore o uguale al 5%.
- Ottimalità: un valore pari a 0.

B.2.1.2 Indice Gulpease

L'Indice Gulpease$_G$ è un indice di leggibilità di un testo tarato sulla lingua italiana, con il vantaggio rispetto ad altri indici di utilizzare la lunghezza delle parole in lettere anzichè in sillabe, semplificandone il calcolo automatico. L'indice utilizza due variabili linguistiche: la lunghezza della parola e la lunghezza della frase rispetto al numero delle lettere.
La formula per il suo calcolo è la seguente:

$$\text{Indice Gulpease}_G = 89 + \frac{300 * (\text{numero delle frasi}) - 10 * (\text{numero delle lettere})}{\text{numero delle parole}}$$

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

- Inferiori 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.

B.2.1.2.1 Soglie

- Accettabilità: un valore superiore o uguale a 45.
- Ottimalità: un valore compreso tra 75 e 100.

B.2.1.3 Errori contenutistici

Questa è la metrica necessaria ad esprimere la correttezza del contenuto di un documento. E' importante verificare che i concetti trattati siano corretti e coerenti con quanto prefissato. Il valore ottenuto da questa metrica rappresenta il numero di errori concettuali che non sono stati corretti dopo esser stati segnalati dal Verificatore$_G$ durante la precedente verifica del documento.
La formula utilizzata per il calcolo degli errori è la seguente:

$$\text{Errori concettuali} = \frac{\text{numero errori non corretti}}{\text{numero totale errori segnalati}} * 100$$

B.2.1.3.1 Soglie

- Accettabilità:un valore inferiore o uguale al 5%.
- Ottimalità: ununun valore uguale allo 0%.

B.2.1.4 Struttura del documento

Viene utilizzata questa unità di misura per verificare quanto un documento sia attinente alle regole strutturali descritte nel documento *Norme di Progetto*. La metrica si basa sul numero di errori segnalati dal *Verificatore* che non sono stati corretti successivamente.
La formula utilizzata per il calcolo degli errori è la seguente:

$$\text{Errori di forma} = \frac{\text{numero errori non corretti}}{\text{numero totale errori segnalati}} * 100$$

B.2.1.4.1 Soglie

- Accettabilità:un valore inferiore o uguale al 5%.
- Ottimalità: ununun valore uguale allo 0%.

B.2.2 Metriche per il prodotto software

In questa sezione si descrivono le metriche che verranno usate dal gruppo per verificare e garantire la qualità dei prodotti software durante il periodo del progetto. Si sottolinea il fatto che questa sarà solo una prima stesura delle metriche e sarà raffinata nel corso delle varie revisioni, facendo frutto dell'esperienza che verrà acquisita negli intervalli di lavoro tra esse.

B.2.2.1 Requisiti soddisfatti

Tale metrica verrà utilizzata per valutare la funzionalità del software prodotto attraverso una misurazione quantificativa dei requisiti soddisfatti; verranno effettuate due misurazioni differenti, una per i soli requisiti obbligatori e una per tutti.

Requisiti obbligatori

$$\text{ROS} = \frac{\text{numero requisiti obbligatori soddisfatti}}{\text{numero totale requisiti obbligatori}}$$

Requisiti obbligatori e facoltativi

$$\text{ROFS} = \frac{\text{numero requisiti obbligatori soddisfatti} + \text{numero requisiti facoltativi soddisfatti}}{\text{numero totale requisiti}}$$

B.2.2.1.1 Soglie

- Accettabilità: il prodotto verrà considerato accettabile quando ROS = 1.
- Ottimalità: il prodotto verrà considerato ottimale quando ROSF = 1.

B.2.2.2 Successo dei test

Tale metrica verrà utilizzata per valutare in parte il livello di affidabilità del prodotto software tramite il calcolo della percentuale di test aventi successo nella fase di verifica.

$$\text{Successo dei test} = \frac{\text{numero test aventi successo}}{\text{numero totale dei test effettuati}} * 100$$

B.2.2.2.1 Soglie

- Accettabilità: valore maggiore o uguale al 98%.
- Ottimalità: valore uguale a 100%; tale risultato non sarà comunque indice di affidabilità totale del software: arrivare ad un tale risultato esigerebbe un carico di lavoro troppo elevato.

B.2.2.3 Tempo di risposta

Tale metrica verrà utilizzata per valutare l'efficienza del prodotto basandosi sul tempo medio che intercorrà tra la richiesta di una certa funzionalità da parte dell'utente e la risposta del software. Con *tempo medio* si intende la media tra i tempi medi di risposta di tutte le funzionalità: ognuna di esse dovrà essere testata almeno 5 volte ed in condizioni quanto più differenti.

$$\mathrm{T}_{\mathrm{rispostaF}} = \frac{\sum_{k=1}^{5} T_{\mathrm{test}} k}{5}$$

$$\mathrm{T}_{\mathrm{rispostaTOT}} = \frac{\sum_{k=1}^{n} T_{\mathrm{rispostaF}} k}{n}$$

B.2.2.3.1 Soglie

- Accettabilità: $\mathrm{T}_{\mathrm{rispostaTOT}}$ compreso tra 0 e 10.
- Ottimalità: $\mathrm{T}_{\mathrm{rispostaTOT}}$ uguale o minore di 1.

B.2.2.4 Validazione pagine web

Tale metrica verrà usata come tentativo di applicare una metrica oggettiva e misurabile per valutare l'usabilità del prodotto finale; si è usata la parola "tentativo" poichè in effetti l'usabilità e l'accessibilità di un sito web sono due cose distinte, anche se affini: pagine web con contenuto inaccessibile sarnno sicuramente poco usabili. Valutare l'accessibilità attraverso l'analisi del codice prodotto permetterà dunque di fornire una base allo sviluppo di pagine usabili. W3C offre uno strumento per valutare le pagine $HTML_{\mathrm{G}}$ e uno per i i fogli di stle CSS_{G}, come dichiarato nelle *Norme di Progetto*: essi riportano il numero e il tipo di errori trovati nei documenti in esame.

B.2.2.4.1 Soglie

- Accettabilità: Saranno accettati file HTML e CSS con un numero di errori minore o uguale a 5 ognuno. In relazione alla dimensione finale del progetto, si darà anche un limite al numero totale degli errori come somma degli errori di tutti i file; si prevedono inoltre modifiche del limite di 5 errori rilevati (aumento o diminuzione) in corso d'opera.
- Ottimalità: Un numero di errori rilevati pari a 0 sarà indice di ottimalità per un file HTML o CSS.

B.2.3 Metriche per il codice

In questa sezione si elencheranno e descriveranno le metriche utilizzare per valutare la qualità del codice sorgente prodotto; la loro applicazione sarà utilizzata per valutare il grado di manutenibilità del prodotto software.

B.2.3.1 Complessità ciclomatica

La complessità ciclomatica è una metrica utilizzata per misurare la complessità di un software attraverso la valutazione dei suoi metodi, classi e algoritmi. Essa è calcolata utilizzando il grafo di flusso: in esso i nodi corrispondono a gruppi indivisibili di istruzioni, mentre gli archi connettono due nodi se le istruzioni del secondo possono essere eseguite immediatamente dopo quelle del primo. Questa misurazione sarà utile nella fase di sviluppo per limitare la complessità delle singole parti del software e nella fase di test per capire quanti test diversi saranno necessari per testare adeguatamente il codice. La misurazione si baserà su un indice numerico intero: valori troppo alti indicano un'eccessiva complessità del codice con conseguente scarsa manutenibilità, mentre valori troppo bassi potrebbero indicare una scarsa efficienza.

B.2.3.1.1 Soglie

- Accettabilità: valore di complessità compreso tra 1 e 15, purchè per valori tra 10 e 15 sia specificato il motivo di tale complessità.
- Ottimalità: valore di complessità compreso tra 1 e 10.

B.2.3.2 Commenti per linee di codice

Attraverso tale metrica si valuterà il rapporto commenti/linee di codice: una percentuale abbastanza alta di commenti aiuterà la comprensione del sorgente. Verrà misurata come segue:

$$\text{CxSLOC} = \frac{\text{Numero linee di commento}}{\text{Numero } SLOC_{\text{G}}} * 100$$

B.2.3.2.1 Soglie

- Accettabilità: sarà accettato un valore CxSLOC compreso tra 20 e 25.
- Ottimalità: sarà dichiarato ottimale un valore CxSLOC compreso tra 25 e 35.

B.2.3.3 Parametri per metodo

Tale metrica si basa sul numero di parametri formali dei $metodi_{\text{G}}$ per valutare la complessità del codice: un numero elevato potrebbe infatti indicare un livello di complessità troppo alto per i singoli metodi.

B.2.3.3.1 Soglie

- Accettabilità: saranno accettati metodi con un numero di parametri minore o uguale a 10.
- Ottimalità: saranno considerati ottimi metodi con un numero di parametri minore o uguale a 5.

B.2.3.4 Linee di codice per metodo

Tale metrica verrà utilizzata congiuntamente alla precedente (Parametri per metodo) per valutare il grado di complessità di un metodo: controllando il numero di $statement_{\text{G}}$ per ogni metodo è possibile facilitarne comprensione e verifica, spingendo verso una modularizzazione del codice il più ampia possibile. Questa metrica sarà fortemente influenzata dall'esperienza che il team guadagnerà durante lo sviluppo del progetto, motivo per cui i valori che seguono saranno indicativi e molto probabilmente modificati in futuro.

B.2.3.4.1 Soglie

- Accettabilità: saranno accettati metodi con una lunghezza pari o inferiore alle 50 righe.
- Ottimalità: saranno considerati ottimi metodi di lunghezza pari o inferiore alle 30 righe.

B.2.3.5 Copertura del codice

Tale metrica è orientata alla valutazione della qualità dei test; essa misura infatti la capacità di coprire mediante test gli statement del codice, attraverso il loro conteggio in percentuale, al fine di fornire dei test che assicurino uan valutazione del software il più affidabile possibile. Verrà calcolata come segue:

$$\text{Copertura} = \frac{\text{Numero di statement testati}}{\text{Numero di statement totali}} * 100$$

B.2.3.5.1 Soglie

- Accettabilità: sarà accettata un numero di statement testati pari al 70%.
- Ottimalità: sarà considerata ottima la capacità di testare almeno il 90% degli statement.

B.2.3.6 Copertura dei branch

Tale metrica verrà utilizzata congiuntamente alla precedente (Copertura del codice) per valutare la qualità dei test. Essa indicherà la capacità dei test di valutare il maggior numero possibile di rami decisionali del grafo di flusso del software. Verrà calcolata come segue:

$$\text{Copertura}_{\text{branch}} = \frac{\text{Numero di rami raggiunti}}{\text{Numero di rami totali}} * 100$$

B.2.3.6.1 Soglie

- Accettabilità: sarà accettata un numero di rami testati pari al 75%.
- Ottimalità: sarà considerata ottima la capacità di testare almeno il 95% dei rami per funzionalità non ancora testate, mentre per codice già testato l'ottimalità sara data dalla capacità di testarne l'80%.

Riferimenti bibliografici

[1] Capitolato d'appalto C6 - Marvin: dimostratore di Uniweb su Ethereum
http://www.math.unipd.it/~tullio/IS-1/2017/Progetto/C6.pdf

[2] Qualità del prodotto software
http://www.math.unipd.it/~tullio/IS-1/2017/Dispense/L13.pdf

[3] Qualità di processo
http://www.math.unipd.it/~tullio/IS-1/2017/Dispense/L15.pdf

[4] ISO/IEC 15504 - SPiCE, Plays-In-Business
http://www.plays-in-business.com/isoiec-15504-spice/

[5] Software Process Improvement and Capability Determination (SPICE) - ISO/IEC 15504
https://shahanali.wordpress.com/2011/04/25/software-process-improvement-and-capability-determinatio

[6] BS ISO/IEC 15504-6:2013

[7] La qualità del software secondo il modello ISO/IEC 9126, Ercole F. Colonese
http://www.colonese.it/00-Manuali_Pubblicatii/07-ISO-IEC9126_v2.pdf

[8] Lo Standard ISO/IEC 9126 – Software engineering – Product Quality, Anna Rita Fasolino
http://www.federica.unina.it/ingegneria/ingegneria-software-ii/
isoiec-9126-software-engineering-quality/

[9] Deming Cycle: The Wheel of Continuous Improvement
https://totalqualitymanagement.wordpress.com/2009/02/25/deming-cycle-the-wheel-of-continuous-improv