



8Lab Solutions - Progetto "Soldino"

## Piano di Qualifica

<b>Versione</b>	1.0.0
<b>Approvazione</b>	
<b>Redazione</b>	
<b>Verifica</b>	
<b>Stato</b>	
<b>Uso</b>	Esterno
<b>Destinato a</b>	8Lab Solutions Prof. Tullio Vardanega Prof. Riccardo Cardin

### **Descrizione**

Questo documento descrive le operazioni di verifica e validazione seguite durante lo svolgimento del progetto Soldino.

8labsolutions@gmail.com

## Tabelle delle modifiche

Versione	Data	Nominativo	Ruolo	Descrizione
0.0.3				.
0.0.2	2018-01-02	Sara Feltrin		Sistemata struttura del documento e inizia stesura §4.
0.0.1	26-12-2018	Paolo Pozzan	<i>Analista</i>	Creato documento L <sup>A</sup> T <sub>E</sub> X e stesura §1.

## Indice

<b>1</b>	<b>Introduzione</b>	<b>6</b>
1.1	Scopo del documento . . . . .	6
1.2	Scopo del prodotto . . . . .	6
1.3	Ambiguità . . . . .	6
1.4	Riferimenti . . . . .	6
1.4.1	Riferimenti normativi . . . . .	6
1.4.2	Riferimenti informativi . . . . .	6
<b>2</b>	<b>Qualità di processo</b>	<b>7</b>
2.1	Processi Primari . . . . .	7
2.1.1	Processo di Analisi dei Requisiti . . . . .	7
2.1.1.1	Obiettivi . . . . .	7
2.1.1.2	Strategia . . . . .	7
2.1.1.3	Metriche . . . . .	7
2.2	Processo di Progettazione Architettuale . . . . .	7
2.2.0.1	Obiettivi . . . . .	7
2.2.0.2	Strategia . . . . .	8
2.2.0.3	Metriche . . . . .	8
2.3	Processo di Progettazione di Dettaglio . . . . .	8
2.3.0.1	Obiettivi . . . . .	8
2.3.0.2	Strategia . . . . .	8
2.3.0.3	Metriche . . . . .	8
2.4	Processi di Supporto . . . . .	8
2.5	Processo di Verifica . . . . .	8
2.5.0.1	Obiettivi . . . . .	9
2.5.0.2	Strategia . . . . .	9
2.5.0.3	Metriche . . . . .	9
2.5.1	Processo di Gestione della Documentazione . . . . .	9
2.5.1.1	Obiettivi . . . . .	9
2.5.1.2	Strategia . . . . .	9
2.5.1.3	Metriche . . . . .	9
2.6	Processi Organizzativi . . . . .	9
2.7	Processo di Pianificazione di Progetto . . . . .	9
2.7.0.1	Obiettivi . . . . .	9
2.7.0.2	Strategia . . . . .	9
2.7.0.3	Metriche . . . . .	9
2.8	Processo di Gestione della Qualità . . . . .	10
2.8.0.1	Obiettivi . . . . .	10
2.8.0.2	Strategia . . . . .	10
2.8.0.3	Metriche . . . . .	10
2.9	Processo di Misurazione . . . . .	10
2.9.0.1	Obiettivi . . . . .	10
2.9.0.2	Strategia . . . . .	10
2.9.0.3	Metriche . . . . .	10

<b>3</b>	<b>Qualità di prodotto</b>	<b>11</b>
3.1	Funzionalità . . . . .	11
3.1.1	Obiettivi . . . . .	11
3.1.2	Metriche . . . . .	11
3.1.2.1	Completezza dell'implementazione . . . . .	11
3.2	Affidabilità . . . . .	11
3.2.1	Obiettivi . . . . .	11
3.2.2	Metriche . . . . .	12
3.2.2.1	Densità errori . . . . .	12
3.3	Usabilità . . . . .	12
3.3.1	Obiettivi . . . . .	12
3.3.2	Metriche . . . . .	12
3.3.2.1	Facilità di utilizzo . . . . .	12
3.3.2.2	Facilità di apprendimento . . . . .	12
3.3.2.3	Profondità della gerarchia . . . . .	12
3.4	Manutenibilità . . . . .	13
3.4.1	Obiettivi . . . . .	13
3.4.2	Metriche . . . . .	13
3.4.2.1	Facilità di comprensione . . . . .	13
3.4.2.2	Semplicità delle funzioni . . . . .	13
3.4.2.3	Semplicità delle classi . . . . .	13
3.5	Metriche per i documenti . . . . .	14
3.5.1	Correttezza ortografica . . . . .	14
3.5.2	Indice di Gulpease . . . . .	14
<b>4</b>	<b>Specifica dei test</b>	<b>15</b>
4.1	Tipi di test . . . . .	15
4.1.1	Test di Unità . . . . .	15
4.1.2	Test di Integrazione . . . . .	15
4.1.3	Test di Sistema . . . . .	15
4.1.4	Test di Accettazione . . . . .	16
<b>A</b>	<b>Standard di qualità</b>	<b>17</b>
A.1	ISO/IEC/IEE 12207:2017 . . . . .	17
A.2	ISO/IEC 9126 . . . . .	18
<b>B</b>	<b>Valutazioni per il miglioramento</b>	<b>20</b>

## Elenco delle figure

## Elenco delle tabelle

# 1 Introduzione

## 1.1 Scopo del documento

Questo documento ha lo scopo di mostrare le strategie di verifica e validazione adottate al fine di garantire la qualità di prodotto e di processo. Per raggiungere questo obiettivo viene applicato un sistema di verifica continua sui processi in corso e sulle attività svolte. In questo modo è quindi possibile rilevare e correggere all'istante eventuali anomalie, riducendo al minimo lo spreco delle risorse.

## 1.2 Scopo del prodotto

Lo scopo del prodotto è quello di realizzare un software, in particolare un sito internet, che consenta il monitoraggio automatico dell'IVA, ovvero assiste il governo e gli utenti nell'esecuzione di operazioni come liquidazione, versamento e rimborso, e permetta l'acquisto di oggetti tramite una valuta denominata "Cubit"<sub>G</sub>.

## 1.3 Ambiguità

All'interno dei documenti alcuni termini presentano significati ambigui a seconda del contesto, fraintendibili, o che necessitano di una descrizione più approfondita. Per evitare questa ambiguità è stato creato il documento "Glossario" volto a fare chiarezza, ponendo a fianco di ogni termine il suo preciso significato. Questi termini sono pertanto marchiati con una "G" a pedice per ogni loro occorrenza all'interno di tutti i documenti.

## 1.4 Riferimenti

### 1.4.1 Riferimenti normativi

- **ISO/IEC 12207**
- **Testo del capitolato**  
<https://www.math.unipd.it/tullio/IS-1/2018/Progetto/C6.pdf>;

### 1.4.2 Riferimenti informativi

- Guide? (vedi Pro-tech)
- **ISO/IEC 9126**  
[https://en.wikipedia.org/wiki/ISO/IEC\\_9126](https://en.wikipedia.org/wiki/ISO/IEC_9126);
- **Indice di Gulpease**  
[https://it.wikipedia.org/wiki/Indice\\_Gulpease](https://it.wikipedia.org/wiki/Indice_Gulpease)

## 2 Qualità di processo

Per ricercare qualità nello svolgimento del progetto si adoperano dei processi. Inizialmente, tali processi sono stati scelti tra quelli proposti nello standard ISO/IEC/IEEE 12207:2017; successivamente sono stati semplificati.

Il risultato sono i processi esposti a seguito.

### 2.1 Processi Primari

#### 2.1.1 Processo di Analisi dei Requisiti

Mediante il processo le informazioni ottenute dalle varie fonti sono trasformate in forma di casi d'uso e requisiti. Questa forma fornisce una descrizione dettagliata del sistema e definisce il funzionamento e le caratteristiche di ogni sua parte.

##### 2.1.1.1 Obiettivi

- definire casi d'uso e requisiti;
- ottenere la loro approvazione;
- tracciare il loro cambiamento nel tempo.

**2.1.1.2 Strategia** Tracciare i requisiti del sistema.

##### 2.1.1.3 Metriche

**Numero di requisiti obbligatori soddisfatti** Indica la percentuale di requisiti obbligatori soddisfatti.

- **Misurazione**
- **Valore preferibile**
- **Valore accettabile**

### 2.2 Processo di Progettazione Architettuale

Tradurre i requisiti in un modello architettuale del sistema. Il modello è ad alto livello di dettaglio: in esso il sistema è costituito da macro-componenti, utili a capire il funzionamento delle parti, ma non ancora realizzabili nella pratica.

Nel processo vanno considerati in particolare il tipo di software da produrre, le caratteristiche desiderate e i suoi requisiti non funzionali per scegliere l'architettura più adatta.

#### 2.2.0.1 Obiettivi

- valutare il tipo di software da produrre, le caratteristiche desiderate, i casi d'uso e i requisiti da soddisfare;
- valutare i modelli architettureali secondo il punto precedente e scegliere il più adeguato;
- individuare nel modello i macro-componenti del sistema;
- comprendere e definire le relazioni tra i componenti.



**2.2.0.2 Strategia** Tracciare le componenti del sistema.

### 2.2.0.3 Metriche

**Structural Fan-in SFIN** Indica quante componenti utilizzano un dato modulo. Un alto valore indica un alto riuso della componente.

- **Misurazione:** numero intero;
- **Valore preferibile:**  $\geq 1$ ;
- **Valore accettabile:**  $\geq 0$ .

**Structural Fan-out SFOUT** Indica quante componenti vengono utilizzate dalla componente in esame. Un alto valore indica un alto accoppiamento della componente.

- **Misurazione:** numero intero;
- **Valore preferibile:**  $= 0$ ;
- **Valore accettabile:**  $\leq 6$ .

## 2.3 Processo di Progettazione di Dettaglio

La progettazione di dettaglio segue la progettazione architetturale, e prevede la scomposizione delle macro-componenti in componenti più piccole che sono:

- immediatamente comprensibili;
- strettamente collegate ai requisiti funzionali;
- sviluppabili da un singolo programmatore.

**2.3.0.1 Obiettivi** Arrivare alla definizione di dettaglio del sistema, cioè scomporlo in unità software dette componenti.

### 2.3.0.2 Strategia

### 2.3.0.3 Metriche

**Numero di metodi per classe**

**Numero di parametri per metodo**

## 2.4 Processi di Supporto

## 2.5 Processo di Verifica

Il processo consiste nella ricerca e correzione di anomalie nei processi e nei prodotti del progetto, mediante tecniche definite.

#### 2.5.0.1 Obiettivi

- individuare e correggere le anomalie;
- provare che il sistema soddisfi i requisiti.

#### 2.5.0.2 Strategia

- individuare tecniche e strumenti di verifica;
- affinare tecniche e strumenti con l'esperienza;

#### 2.5.0.3 Metriche

**Code Coverage** Indica il numero di righe di codice percorse dai test durante la loro esecuzione.

- misurazione:  $Le/Lt$ ;
- valore preferibile: 100%;
- valore accettabile: 75%.

### 2.5.1 Processo di Gestione della Documentazione

#### 2.5.1.1 Obiettivi

#### 2.5.1.2 Strategia

#### 2.5.1.3 Metriche

- 
- 
- 

## 2.6 Processi Organizzativi

## 2.7 Processo di Pianificazione di Progetto

#### 2.7.0.1 Obiettivi

#### 2.7.0.2 Strategia

#### 2.7.0.3 Metriche

- 
- 
- 

## **2.8 Processo di Gestione della Qualità**

### **2.8.0.1 Obiettivi**

### **2.8.0.2 Strategia**

### **2.8.0.3 Metriche**

- 
- 
- 

## **2.9 Processo di Misurazione**

**2.9.0.1 Obiettivi** Scegliere metriche adeguate, effettuare misurazioni basate su tali metriche per migliorare, controllare e garantire la qualità di prodotto e processo.

### **2.9.0.2 Strategia**

- definire gli oggetti di interesse da misurare;
- scegliere metriche appropriate;
- effettuare le misurazioni nel modo più possibile automatizzato;

### **2.9.0.3 Metriche**

- 
- 
-

### 3 Qualità di prodotto

Per valutare la qualità del prodotto il gruppo ha deciso di far riferimento allo standard ISO/IEC 9126<sub>G</sub> che definisce le caratteristiche di cui tener conto nel momento in cui si voglia produrre un prodotto di buona qualità. Le caratteristiche sono descritte attraverso dei parametri misurabili che permettono di quantificare il raggiungimento della caratteristica in questione. Di seguito vengono citate le voci che il gruppo ha ritenuto rilevanti in relazione al contesto.

#### 3.1 Funzionalità

Capacità del prodotto di fornire funzioni che riescano a soddisfare tutti i requisiti, sia quelli espliciti che quelli impliciti, presenti nell'Analisi dei Requisiti.

##### 3.1.1 Obiettivi

- **Appropriatezza:** il prodotto deve mettere a disposizione un insieme di funzioni conformi agli obiettivi richiesti;
- **Accuratezza:** il prodotto deve fornire risultati attestati con il grado di precisione richiesto;
- **Conformità:** il prodotto deve aderire a determinati standard.

##### 3.1.2 Metriche

###### 3.1.2.1 Completezza dell'implementazione

- La completezza del prodotto e il rispetto dei requisiti viene indicato da una percentuale che si calcola con la seguente formula:

$$C = (1 - \frac{N_{FNI}}{N_{FI}}) \cdot 100$$

dove  $N_{FNI}$  indica il numero di funzionalità non implementate e  $N_{FI}$  indica il numero di funzionalità individuate dall'analisi.

- Il valore di accettazione di  $C$  è 100%, a indicare quindi che tutte le funzionalità, individuate dai requisiti obbligatori, sono state implementate

#### 3.2 Affidabilità

Capacità del prodotto di mantenere prestazioni elevate anche in caso di situazioni non normali o critiche.

##### 3.2.1 Obiettivi

- **Maturità:** il prodotto deve evitare che si verifichino errori e malfunzionamenti;
- **Tolleranza agli errori:** il prodotto mantiene alte prestazioni anche in caso di malfunzionamenti o di un uso scorretto.

### 3.2.2 Metriche

#### 3.2.2.1 Densità errori

- L'abilità del prodotto di resistere a malfunzionamenti viene indicata con una percentuale che si calcola con la seguente formula:

$$M = \frac{N_{ER}}{N_{TE}} \cdot 100$$

dove  $N_{ER}$  indica il numero di errori rilevati durante il testing e  $N_{TE}$  indica il numero di test eseguiti;

- Il valore preferibile è 0%;
- Un valore accettabile è non superiore al 10%.

### 3.3 Usabilità

Capacità del prodotto di essere di facile comprensione e utilizzo da parte degli utenti.

#### 3.3.1 Obiettivi

- **Comprensibilità:** l'utente deve essere in grado di comprendere le funzionalità offerte dal prodotto e ad utilizzarle;
- **Apprendibilità:** l'utente deve poter imparare facilmente ad utilizzare il prodotto;
- **Attrattività:** il prodotto deve essere piacevole all'utente che ne fa utilizzo;

#### 3.3.2 Metriche

##### 3.3.2.1 Facilità di utilizzo

- La facilità con cui l'utente raggiunge ciò che vuole viene rappresentata tramite il numero di click che sono necessari per arrivare ad esempio alla pagina di fine ordine;
- Il valore preferibile è 10 click;
- Un valore accettabile è non superiore a 15.

##### 3.3.2.2 Facilità di apprendimento

- La facilità con cui l'utente riesce ad imparare ad usare le funzionalità del prodotto viene rappresentata tramite il tempo medio che serve per comprenderle;
- Il valore preferibile è 3 minuti;
- Un valore accettabile è non superiori a 5.

##### 3.3.2.3 Profondità della gerarchia

- La profondità del sito. Un sito per essere facile da utilizzare non deve essere troppo profondo;
- Il valore preferibile è 4;
- Un valore accettabile è inferiore a 7.

### 3.4 Manutenibilità

Capacità del prodotto di essere modificato, includendo correzioni, miglioramenti o adattamenti.

#### 3.4.1 Obiettivi

- **Analizzabilità:** facilità con la quale è possibile analizzare il codice per localizzare un errore;
- **Modificabilità:** capacità del prodotto di permettere l'implementazione di una modifica;

#### 3.4.2 Metriche

##### 3.4.2.1 Facilità di comprensione

- La facilità con cui è possibile comprendere cosa fa il codice può rappresentata dalla seguente formula:

$$R = \frac{N_{LCOM}}{N_{LCOD}}$$

dove  $N_{LCOM}$  indica le linee di commento e  $N_{LCOD}$  indica le linee di codice;

- Il valore ottimale di  $R$  è superiore a 0.20;
- Un valore accettabile è superiore a 0.10.

##### 3.4.2.2 Semplicità delle funzioni

- La facilità di un metodo può essere rappresentata dal numero di parametri per metodo: meno parametri ha una funzione più è semplice e intuitiva;
- Il valore preferibile è inferiore o uguale ai 3 parametri per metodo;
- Un valore accettabile è non superiore a 6.

##### 3.4.2.3 Semplicità delle classi

- La facilità di una classe può essere rappresentata dal numero di metodi per classe: una classe con pochi metodi ha uno scopo ben preciso e facilmente comprensibile;
- Il valore ottimale di metodi per classe è inferiore a 8;
- Un valore accettabile è non superiore a 15.

### 3.5 Metriche per i documenti

Per assicurare che tutti i documenti prodotti siano leggibili, comprensibili e corretti sotto tutti i punti di vista questi devono rispettare determinate metriche.

#### 3.5.1 Correttezza ortografica

Tutti i documenti, sia quelli in italiano che quelli in inglese, non devono assolutamente contenere errori grammaticali o errori ortografici.

- Saranno accettati solamente documenti privi di errori grammaticali o ortografici.

#### 3.5.2 Indice di Gulpease

Il gruppo si impegna a scrivere documenti facilmente leggibili.

- L'indice di Gulpease fornisce una misura della leggibilità di un documento scritto italiano. La formula che calcola questo indice è la seguente:

$$89 + \frac{330 \cdot (\text{numero delle frasi}) - 10 \cdot (\text{numero delle lettere})}{\text{numero delle parole}}$$

Questa formula produce un risultato compreso tra 0 e 100 dove 100 indica la leggibilità più alta;

- Il valore ottimale sarà superiore a 80;
- Un valore accettabile è superiore a 40.

## 4 Specifica dei test

Per produrre software di qualità, il gruppo *8Lab Solutions* definirà dei test per assicurarsi che le unità prodotte funzionino in maniera corretta. Seguirà quindi il tracciamento dei test e il loro esito per mezzo di tabelle che ne semplificheranno la consultazione e che potranno fornire una precisa indicazione degli output prodotti, specificando se il risultato ottenuto sia quello atteso, errato oppure non coerente a quanto fissato in precedenza. Per definire lo stato dei test, vengono utilizzate le seguenti sigle:

- **I**: per indicare che il test è stato implementato;
- **NI**: per indicare che il test non è stato implementato.

Inoltre per lo stato dei test si usano le seguenti abbreviazioni:

- **S**: per indicare che il test ha soddisfatto la richiesta;
- **NS**: per indicare che il test non ha soddisfatto la richiesta.

### 4.1 Tipi di test

Vengono individuate quattro tipologie di test:

- **Test di Unità [TU]**: questi test hanno il compito di verificare le singole unità del software, ovvero le minime componenti del programma che hanno un funzionamento autonomo. Il successo da parte di questi test non implica il corretto funzionamento da parte del software;
- **Test di Integrazione [TI]**: lo scopo di questi test è quello di testare come un gruppo i singoli moduli del software. Essi vengono svolti dopo i Test di Unità e prima dei Test di Sistema;
- **Test di Sistema [TS]**: questi test vengono utilizzati quando il sistema viene installato su una piattaforma e verificano che esso raggiunga gli obiettivi fissati e soddisfi le richieste formulate in partenza;
- **Test di Accettazione [TA]**: essi vengono fatti alla fine per verificare che il prodotto finale soddisfi quanto richiesto dal proponente.

#### 4.1.1 Test di Unità

I test di unità hanno l'obiettivo di verificare il corretto funzionamento della parte più piccola autonoma del lavoro realizzato.

#### 4.1.2 Test di Integrazione

I test di integrazione sono usati per verificare il corretto funzionamento tra le varie unità dell'architettura.

#### 4.1.3 Test di Sistema

I test di sistema sono impiegati per garantire il corretto funzionamento delle componenti dell'intero sistema.



#### **4.1.4 Test di Accettazione**

I test di accettazione hanno lo scopo di dimostrare che il software sviluppato soddisfi le richieste del proponente ed essi vengono eseguiti durante il collaudo finale.

## **A Standard di qualità**

### **A.1 ISO/IEC/IEEE 12207:2017**

## A.2 ISO/IEC 9126

ISO/IEC 9126 è uno standard internazionale per valutare la qualità del software.

Questo standard è diviso in quattro parti:

- **Modello della qualità del software** (descritto dopo le successive 3 parti)
- **Metriche per la qualità interna:** metriche che si applicano al software non eseguibile durante le fasi di progettazione e codifica. Permettono di individuare eventuali problemi che potrebbero influire sulla qualità finale del prodotto prima che venga realizzato un eseguibile. Grazie alle misure effettuate tramite le metriche interne è possibile prevedere il livello di qualità esterna e di qualità in uso del prodotto finale, poiché entrambe vengono influenzate dalla qualità interna.  
Viene rilevata tramite analisi statica. Idealmente la qualità interna determina la qualità esterna;
- **Metriche per la qualità esterna:** metriche applicabili al software in esecuzione che ne misurano il comportamento attraverso dei test, in funzione degli obiettivi stabiliti.  
Viene rilevata tramite analisi dinamica. Idealmente la qualità esterna determina la qualità in uso;
- **Metriche per la qualità in uso:** metriche applicabili solo al prodotto finito ed in uso in condizioni reali.  
La qualità in uso viene raggiunta solo se è stato raggiunto il livello di qualità interna e di qualità esterna.

Il modello di qualità del software, presentato nella prima parte dello standard, suddivide la qualità in sei caratteristiche generali e varie sotto caratteristiche, misurabili attraverso delle metriche, utilizzate per fornire una scala ed un metodo per la misurazione. Ecco l'elenco delle caratteristiche:

- **Funzionalità:** capacità del software di soddisfare i requisiti, descritti nell'Analisi dei Requisiti, in un determinato contesto.  
Nello specifico il software deve soddisfare le seguenti caratteristiche:
  - **Appropriatezza:** capacità di fornire funzioni appropriate per attività specifiche, che permettano di raggiungere gli obiettivi prefissati;
  - **Accuratezza:** capacità di fornire i risultati concordati o la precisione richiesta;
  - **Interoperabilità:** capacità di interagire ed operare con uno o più sistemi specificati;
  - **Conformità:** capacità di aderire a standard;
  - **Sicurezza:** capacità di proteggere informazioni e dati.
- **Affidabilità:** capacità del software di mantenere uno specifico livello di prestazioni quando usato in condizioni specificate.  
Nello specifico il software deve soddisfare le seguenti caratteristiche:
  - **Maturità:** capacità di evitare il verificarsi di errori, malfunzionamenti o risultati non corretti;
  - **Tolleranza agli errori:** capacità di mantenere livelli prefissati di prestazioni anche in presenza di malfunzionamenti o usi scorretti del prodotto finale;

- **Recuperabilità:** capacità di ripristinare un livello appropriato di prestazioni o di recupero di informazioni rilevanti a seguito di un malfunzionamento;
- **Aderenza:** capacità di aderire a standard, regole e convenzioni che riguardano l'affidabilità.
- **Efficienza:** capacità del prodotto software di eseguire le proprie funzioni minimizzando il tempo necessario e sfruttando al meglio le risorse che necessita.  
Nello specifico il software deve soddisfare le seguenti caratteristiche:
  - **Nel tempo:** capacità di fornire adeguati tempi di risposta, elaborazione e velocità di attraversamento in determinate condizioni;
  - **Nello spazio:** capacità di utilizzo di quantità e tipo di risorse in maniera adeguata;
- **Usabilità:** capacità del prodotto software di essere compreso, appreso, usato e accettato dall'utente, quando usato sotto determinate condizioni.  
Nello specifico il software deve soddisfare le seguenti caratteristiche:
  - **Comprensibilità:** capacità di essere chiaro riguardo le proprie funzionalità e il proprio utilizzo;
  - **Apprendibilità:** capacità di essere facilmente apprendibile dagli utenti;
  - **Operabilità:** capacità di permettere all'utente di eseguire i suoi scopi e controllarne l'uso;
  - **Attrattività:** capacità di essere piacevole all'utente che l'utilizza.
- **Manutenibilità:** capacità del software di essere modificato, al fine di aggiungere correzioni, miglioramenti o adattamenti.  
Nello specifico il software deve soddisfare le seguenti caratteristiche:
  - **Analizzabilità:** capacità di essere facilmente analizzato al fine di localizzare un errore;
  - **Modificabilità:** capacità di poter essere agevolmente modificato nel codice, nella progettazione o nella documentazione;
  - **Stabilità:** capacità di evitare effetti indesiderati a seguito di una modifica;
  - **Testabilità:** capacità di essere facilmente testato per validare le modifiche apportate.
- **Portabilità:** capacità del software di essere trasportato da un ambiente di lavoro ad un altro, sia esso hardware che software.  
Nello specifico il software deve soddisfare le seguenti caratteristiche:
  - **Adattabilità:** capacità di essere facilmente adattato a differenti ambienti operativi, senza applicare modifiche;
  - **Installabilità:** capacità di poter essere installato in un determinato ambiente;
  - **Conformità:** capacità di coesistere con altre applicazioni e di condividere risorse;
  - **Sostituibilità:** capacità di essere utilizzato al posto di un altro software per svolgere gli stessi compiti, nello stesso ambiente.

## B Valutazioni per il miglioramento