8Lab Solutions - Progetto "Soldino"

Piano di Qualifica

Versione	1.0.0
Approvazione	
Redazione	
Verifica	
Stato	
Uso	Esterno
Destinato a	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	*Analista*	Creato documento LaTeX e stesura §1.

Indice

Elenco delle figure

Elenco delle tabelle

Introduzione

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.

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"$_G$.

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.

Riferimenti

Riferimenti normativi

- **ISO/IEC 12207**

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

Riferimenti informativi

- **ISO/IEC 9126**
  https://en.wikipedia.org/wiki/ISO/IEC_9126;
- **Indice di Gulpease**
  https://it.wikipedia.org/wiki/Indice_Gulpease
- **Schedule Variance e metriche correlate**
  https://www.smartsheet.com/hacking-pmp-how-calculate-schedule-variance

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.

Processi Primari

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

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

2.1.1.2 Strategia

- considerare lo scopo del progetto e le richieste degli stakeholder;
- esprimere ciò in forma di requisiti, classificati in obbligatori, desiderabili e opzionali;
- valutare il corpo dei requisiti e negoziare cambiamenti se necessario;
- ottenere la loro approvazione da parte del proponente;

Disporre del tracciamento dei requisiti del sistema.

2.1.1.3 Metriche

Percentuale di requisiti obbligatori soddisfatti - PROS Indica appunto la percentuale di requisiti obbligatori soddisfatti.

- misurazione: valore percentuale: $PROS = \frac{requisitiobbligatorisoddisfatti}{requisitiobbligatoritotali}$;
- valore preferibile: 100%;
- valore accettabile: 100%.

Processo di Definizione dell'Architettura

Tradurre i requisiti in un modello architetturale 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 architetturali 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**: valore intero: conteggio delle componenti;
- **Valore preferibile**: $SFIN \geq 1$;
- **Valore accettabile**: $SFIN \geq 0$.

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

- **Misurazione**: valore intero: conteggio delle componenti;
- **Valore preferibile**: $SFOUT = 0$;
- **Valore accettabile**: $SFOUT \leq 6$.
- **Misurazione**: numero intero;
- **Valore preferibile**: $= 0$;
- **Valore accettabile**: ≤ 6.

Processo di Progettazione di Design

La Progettazione di Design segue la Definizione dell'Architettura 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 Ottenere unità software atomiche dette componenti, facili da tradurre in codice e da testare.

2.3.0.2 Strategia

2.3.0.3 Metriche

Numero di metodi per classe

Numero di parametri per metodo

Processi di Supporto

Processo di Verifica

Il processo consiste nella ricerca e correzione di anomalie e difetti nei processi e prodotti del progetto, mediante tecniche predefinite e se possibile automatiche.

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;
- applicarli;
- affinarli con l'esperienza;

2.5.0.3 Metriche

Code Coverage - CC Indica il numero di righe di codice percorse dai test durante la loro esecuzione. Per linee di codice totali si intende tutte quelle appartenenti all'unità in fase di test.

- misurazione: valore percentuale: $CC = \frac{linee\ di\ codice\ percorse}{linee\ di\ codice\ totali}$;
- valore preferibile: 100%;
- valore accettabile: 75%.

Processo di Gestione dell'Informazione

Il processo consiste nella produzione di informazioni, la cui forma tipica è documentale, e nella loro gestione. I documenti sono prodotti a supporto di tutte le attività di progetto.

2.5.1.1 Obiettivi Si costruisce la documentazione affinché sia un body of knowledge che raccoglie la conoscenza in modo:

- completo;
- non ambiguo;
- modulare e fatto di parti coese;
- trasparente, adatto alla trasmissione delle informazioni;
- disponibile esternamente.

2.5.1.2 Strategia

I documenti sono:

- prodotti in concomitanza con tutte le attività di sviluppo;
- prodotti in modo collaborativo;
- supportati da un glossario;
- supportati da Norme di Progetto;
- prodotti con strumenti software adatti alla collaborazione e alla modularità in LaTeX;
- ospitati in una repository pubblica su Github;

2.5.1.3 Metriche

Indice di Gulpease Indice della leggibilità del testo. Valuta la lunghezza delle parole e delle frasi rispetto al numero totale di lettere.

- misurazione: valore intero da 0 a 100:
  $I_G = 89 + \frac{(300 * numero\ di\ frasi - 10 * numero\ di\ lettere)}{numero\ di\ parole}$
- valore preferibile: $80 < I_G < 100$;
- valore accettabile: $40 < I_G < 100$

Processi Organizzativi

Processo di Pianificazione di Progetto

Il processo consiste nella produzione di un Piano di Progetto e nella sua manutenzione. Il piano governa le risorse a disposizione, ovvero tempi, costi e ruoli.

2.7.0.1 Obiettivi

- produrre la pianificazione delle attività;
- mantenerla aggiornata mentre esse vengono svolte;
- usarla come riferimento e supporto.

2.7.0.2 Strategia

2.7.0.3 Metriche

BAC: Budget at Completion Budget totale allocato per il progetto

- misurazione: numero intero;
- valore preferibile: *parialpreventivo*;
- valore accettabile: *dipocomaggioreoparialpreventivo*.

AC: Actual Cost Il denaro speso fino al momento del calcolo.

- misurazione: numero intero;
- valore preferibile: $0 \leq AC < PV$;
- valore accettabile: $0 \leq AC \leq budgettotale$;

EV - Earned Value Metrica di utilità. Si tratta del valore del lavoro fatto fino al momento del calcolo; corrisponde al denaro guadagnato fino a quel momento.

- misurazione: $BAC * \%dilavorocompletato$;
- valore preferibile: $EV \geq 0$;
- valore accettabile: $EV \geq 0$.

PV: Planned Value Si tratta del valore del lavoro pianificato al momento del calcolo: corrisponde al denaro che si dovrebbe aver guadagnato in quel momento.

- misurazione: $BAC * \%dilavoropianificato$;
- valore preferibile: $PV \geq 0$;
- valore accettabile: $PV \geq 0$;

SV: Schedule Variance Dice se si è avanti o indietro nello svolgimento del progetto rispetto alla pianificazione.

- misurazione: $SV = EV - PV$
- valore preferibile: $SV > 0$;
- valore accettabile: $SV = 0$;

Cost Variance Differenza tra il costo del lavoro effettivamente completato e quello pianificato. Una CV positiva indica che si sta rispettando il budget.

- misurazione: $CV = EV - AC$;
- valore preferibile: $CV > 0$
- valore accettabile: $CV \geq 0$

Processo di Gestione della Qualità

Tale processo ha per scopo il raggiungimento di un grado soddisfacente di qualità nel progetto. Fornisce:

- obiettivi da perseguire;
- strumenti tecnici, come procedure e metriche.

2.8.0.1 Obiettivi Garantire che i prodotti e processi rispettino gli standard di qualità richiesti.

2.8.0.2 Strategia

- pianificare le proprie azioni perseguendo gli obiettivi;
- utilizzare gli strumenti forniti per misurare e monitorare i risultati;
- reagire ai risultati aggiornando obiettivi, strategia e strumenti.

2.8.0.3 Metriche

Percentuale di metriche soddisfatte - PMS La percentuale di metriche soddisfatte valuta quante metriche raggiungono soglie accettabili sul numero totale delle metriche calcolate. Una bassa percentuale di soddisfazione può indicare poca qualità, metriche inadeguate o mancata correttezza nel calcolo.

- misurazione: $\frac{numero\ di\ metriche\ soddisfatte}{numero\ di\ metriche\ totali}$;
- valore preferibile: $PMS \geq 80\%$;
- valore accettabile: $PMS \geq 60\%$;

Qualità di prodotto

Per valutare la qualità del prodotto il gruppo ha deciso di far riferimento allo standard ISO/IEC 9126$_G$ 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 del progetto.

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.

Obiettivi

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

Metriche

3.1.2.1 Completezza dell'implementazione La completezza del prodotto e il rispetto dei requisiti viene indicato da una percentuale.

- **Misurazione**: 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;
- **Valore accettabile**: 100%;
- **Valore accettabile**: 100%.

Affidabilità

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

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.

Metriche

3.2.2.1 Densità errori L'abilità del prodotto di resistere a malfunzionamenti viene indicata con una percentuale.

- **Misurazione**: 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;
- **Valore preferibile**: 0%;
- **Valore accettabile**: $\leq$ 10%.

Usabilità

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

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.

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 al contenuto desiderato.

- **Misurazione**: Click per raggiungere la schermata di checkout;
- **Valore preferibile**: 10;
- **Valore accettabile**: $\leq$ 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.

- **Misurazione**: Minuti per raggiungere pagina di checkout;
- **Valore preferibile** 3;
- **Valore accettabile** $\leq$ 5.

3.3.2.3 Profondità della gerarchia La profondità del sito. Un sito sito per essere facile da utilizzare non deve essere troppo profondo.

- **Misurazione**: Livello di profondità della pagina di un oggetto;
- **Valore preferibile**: 4;
- **Valore accettabile**: $\leq$ 7.

Manutenibilità

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

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.

Metriche

3.4.2.1 Facilità di comprensione La facilità con cui è possibile comprendere cosa fa il codice può rappresentata dal numero di linee di commento nel codice.

- **Misurazione**: Si può calcolare con la seguente formula:
$$\mathrm{R} = \frac{N_{\mathrm{LCOM}}}{N_{\mathrm{LCOD}}}$$
dove $\mathrm{N_{LCOM}}$ indica le linee di commento e $\mathrm{N_{LCOD}}$ indica le linee di codice;
- **Valore preferibile**: ≥ 0.20;
- **Valore accettabile**: ≥ 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.

- **Misurazione**: Numero di parametri per metodo;
- **Valore preferibile** ≤ 3;
- **Valore accettabile** ≤ 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.

- **Misurazione**: Numero di metodi per classe;
- **Valore preferibile** ≤ 8;
- **Valore accettabile** ≤ 15.

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.

Correttezza ortografica

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

- **Misurazione**:Numero di errori grammaticali o ortografici;
- **Valore preferibile** 0;
- **Valore accettabile** 0.

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.

- **Misurazione**: La formula che calcola questo indice è la seguente:
  $$89 + \frac{330 \cdot (numero\,delle\,frasi) - 10 \cdot (numero\,delle\,lettere)}{numero\,delle\,parole}$$
  Questa formula produce un risultato compreso tra 0 e 100 dove 100 indica la leggibilità più alta;
- **Valore preferibile**: ≥ 80;
- **Valore accettabile**: ≥ 40.

Specifica dei test

Per assicurare la qualità del software prodotto, il gruppo *8Lab Solutions* adotterà come modello si sviluppo del software il **Modello a V**, il quale prevede lo sviluppo dei test in parallelo alle attività di analisi e progettazione. In questo modo i test permetteranno di verificare sia la correttezza delle parti di programma sviluppati, sia che tutti gli aspetti del progetto siano implementati e corretti. 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.

Tipi di test

Vengono individuate quattro tipologie di test:

- **Test di Accettazione [TA]**: essi vengono fatti alla fine per verificare che il prodotto finale soddisfi quanto richiesto dal proponente;
- **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 Integrazione [TI]**: lo scopo di questi test è quello di testare come un gruppo i singoli moduli$_G$ del software. Essi vengono svolti dopo i Test di Unità e prima dei Test di Sistema;
- **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 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.

Test di Sistema

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

Test di Integrazione

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

Test di Unità

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

Standard di qualità

ISO/IEC/IEE 12207:2017

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.

Valutazioni per il miglioramento

L'obiettivo di questa appendice è la valutazione atta a migliorare l'intero processo produttivo legato al progetto in corso. Risulta quindi necessario trovare un modo per affrontare i problemi che possono sorgere durante il lavoro, al fine di dare soluzioni efficienti alla risoluzione degli stessi e evitare così che si ripetano.
Verano dunque tracciati problemi riguardanti i seguenti ambiti:

- **Organizzazione**: Problemi riguardanti l'organizzazione e la comunicazione all'interno del gruppo:
- **Ruoli**: problemi riguardanti il corretto svolgimento di un ruolo;
- **Strumenti**: problemi riguardanti l'uso degli strumenti scelti.

Al fine di evitare di incappare più volte nello stesso problema ripetutamente, si è deciso di scrivere questa appendice in cui vengono registrati i problemi riscontrati e le soluzioni adottate. Ogni problema viene sollevato sulla base dell'autovalutazione dei membri del gruppo, a causa della mancanza di una persona esterna al gruppo che dia una valutazione oggettiva. Nonostante sia un sistema meno efficace può comunque aiutare a risolvere problemi difficili per il singolo, evitare di commettere lo stesso errore più volte e migliorare progressivamente la qualità del lavoro.
Questa sezione non contiene tutti i problemi che saranno incontrati durante lo svolgimento del progetto, verrà quindi aggiornata man mano che questi verranno incontrati.

Valutazioni sull'organizzazione

- Si è rivelato difficoltoso organizzare incontri nei quali fossero presenti tutti i membri, si è scelto quindi di organizzare incontri più frequentemente anche se non tutti i membri sono presenti;

Valutazione sui ruoli

- Il responsabile, a causa dell'inesperienza, ha trovato difficoltoso suddividere in modo bilanciato il lavoro tra i membri, si è scelto quindi di ridurre il carico assegnato ad ogni singolo membro;

Valutazioni sugli strumenti

- Si è verificato difficoltoso per alcuni membri, causa inesperienza, la stesura dei documenti LaTeXsoprattutto per quanto riguarda la costruzione di tabelle e l'inserimento di figure, si è scelto quindi di far creare ai membri più esperti template e comandi personalizzati.
- Si sono verificati alcuni conflitti durante il push di modifiche su Github, si è scelto quindi di togliere la possibilità di pushare direttamente sul master senza che almeno un altro membro approvati le modifiche.