

roundabout

Roundabout - Etherless

Piano di Qualifica

Versione	0.1.0
Approvazione	
Redazione	Luca Benetazzo Nicoletta Fabro Feim Jakupi
Verifica	Antonio Zlatkovski
Stato	Non approvato
Uso	Esterno
Destinato a	Roundabout Prof. Tullio Vardanega Prof. Riccardo Cardin

Descrizione

Questo documento descrive le operazioni di verifica e validazione seguite dal gruppo Roundabout per il progetto Etherless

`team.roundabout.13@gmail.com`

Registro delle modifiche

Versione	Data	Nominativo	Ruolo	Descrizione
0.1.0	2020-04-02	Antonio Zlatkovski	<i>Verificatore</i>	Verifica §1, §2, §4, §5 e sezione B.
0.0.7	2020-04-02	Luca Benetazzo	<i>Verificatore</i>	Stesura §4 e §5.
0.0.6	2020-04-01	Luca Benetazzo	<i>Verificatore</i>	Stesura §C.
0.0.5	2020-04-01	Nicoletta Fabro	<i>Progettista</i>	Stesura §2.1, §2.2, §2.3.
0.0.4	2020-03-30	Luca Benetazzo	<i>Verificatore</i>	Stesura §B.
0.0.3	2020-03-26	Luca Benetazzo	<i>Verificatore</i>	Stesura §1.
0.0.2	2020-03-21	Nicoletta Fabro	<i>Progettista</i>	Organizzazione struttura documento.
0.0.1	2020-03-20	Luca Benetazzo	<i>Amministratore</i>	Creazione documento \LaTeX .

Indice

1	Introduzione	5
1.1	Premessa	5
1.2	Scopo del documento	5
1.3	Scopo del prodotto	5
1.4	Glossario	5
1.5	Riferimenti	5
1.5.1	Riferimenti normativi	5
1.5.2	Riferimenti informativi	5
2	Qualità di Processo	7
2.1	Scopo	7
2.2	Processi Primari	7
2.2.1	Processi di sviluppo	7
2.2.1.1	Analisi dei Requisiti	7
2.2.1.2	Progettazione	7
2.2.1.3	Codifica	8
2.3	Processi di Supporto	9
2.3.0.1	Pianificazione	9
2.3.0.2	Verifica	11
2.3.0.3	Documentazione	11
2.4	Processi Organizzativi	11
2.4.0.1	Gestione della Qualità	11
3	Qualità di Prodotto	13
3.1	Scopo	13
3.2	Funzionalità	13
3.3	Affidabilità	13
3.4	Usabilità	13
3.5	Manutenibilità	13
4	Test di Verifica	14
4.1	Test di Unità	14
4.2	Test di Integrazione	14
4.3	Test di Sistema	14
4.4	Test di Regression	14
5	Test di Validazione	15
5.1	Test di Accettazione	15
5.1.1	Test funzionali	15
5.1.2	Test di qualità	21
5.1.3	Test di vincolo	22
A	Standard di qualità	24
A.1	ISO/IEC 9126	24
A.1.1	Modello per la qualità del software	24
A.1.1.1	Modello per la qualità esterna ed interna	24
A.1.1.2	Modello per la qualità in uso	26

A.1.2	Metriche per la qualità interna	26
A.1.3	Metriche per la qualità esterna	27
A.1.4	Metrica per la qualità in uso	27
A.2	ISO/IEC 15504	27
A.3	Ciclo di Deming	27
B	Valutazioni per il miglioramento	28
B.1	Valutazioni sull'organizzazione	28
B.2	Valutazioni sui ruoli	28
B.3	Valutazioni sugli strumenti di lavoro	29
C	Resoconto delle attività di verifica	30
C.1	Verifiche statiche	30
C.2	Verifiche requisiti	30
C.3	Verifiche automatizzate	30

Elenco delle figure

A.1.1Modello ISO/IEC 9126	24
-------------------------------------	----

1 Introduzione

1.1 Premessa

Il *Piano di Qualifica* è un documento di cui si prevede la stesura durante l'intera durata del progetto, adottando una modalità incrementale. Per questo motivo, non è da considerarsi equivalente ad un documento completo.

1.2 Scopo del documento

Questo documento contiene tutte le strategie di verifica e validazione adottate dal gruppo *Roundabout*, al fine di garantire la qualità di prodotto e processo. Per ottenere questo obiettivo viene applicato una verifica continua sui processi in corso e sulle attività svolte. Procedendo in questo modo si eviteranno più facilmente eventuali malformità e si consentirà una manutenzione qualitativamente migliore.

1.3 Scopo del prodotto

L'applicativo che si vuole sviluppare è *Etherless*, una piattaforma cloud che sfrutta la tecnologia degli smart contract caratteristica della rete Ethereum_G. Lo scopo di *Etherless* è duplice: da una parte permettere agli *sviluppatori* di rilasciare funzioni Javascript_G nel cloud_G, dall'altra permettere agli *utenti* di beneficiare di queste funzioni in seguito ad un pagamento per il loro uso. *Etherless* è gestita e mantenuta dai suoi *amministratori*.

1.4 Glossario

Al fine di evitare possibili ambiguità, i termini tecnici utilizzati nei documenti formali vengono chiariti ed approfonditi nel *Glossario Interno 1.0.0*. Per facilitare la lettura, i termini presenti in tale documento sono contrassegnati in tutto il resto della documentazione da una 'G' a pedice.

1.5 Riferimenti

1.5.1 Riferimenti normativi

- **Norme di Progetto:** *Norme di Progetto v1.0.0*;
- **Capitolato d'appalto C2 - Etherless:**
<https://www.math.unipd.it/~tullio/IS-1/2019/Progetto/C2.pdf>.

1.5.2 Riferimenti informativi

- **Standard ISO/IEC 9126:**
https://it.wikipedia.org/wiki/ISO/IEC_9126;
- **Standard ISO/IEC 15504:**
https://en.wikipedia.org/wiki/ISO/IEC_15504;
- **Ciclo di Deming:**
https://it.wikipedia.org/wiki/Ciclo_di_Deming;
- **Indice di Gulpease:**
https://it.wikipedia.org/wiki/Indice_Gulpease;

- **Slide Qualità di prodotto:**
<https://www.math.unipd.it/~tullio/IS-1/2019/Dispense/L12.pdf>;
- **Slide Qualità di processo:**
<https://www.math.unipd.it/~tullio/IS-1/2019/Dispense/L13.pdf>;
- **Slide Verifica e Validazione:**
 - <https://www.math.unipd.it/~tullio/IS-1/2019/Dispense/L14.pdf>;
 - <https://www.math.unipd.it/~tullio/IS-1/2019/Dispense/L15.pdf>;
 - <https://www.math.unipd.it/~tullio/IS-1/2019/Dispense/L16.pdf>;

2 Qualità di Processo

2.1 Scopo

Lo scopo della seguente sezione è quello di illustrare le metriche adottate per garantire qualità nello svolgimento dei processi, precedentemente scelti tra quelli proposti dallo standard ISO/IEC 12207:1995 ed illustrati nel documento *Norme di Progetto*.

2.2 Processi Primari

2.2.1 Processi di sviluppo

2.2.1.1 Analisi dei Requisiti

Le metriche usate per l'attività di analisi sono le seguenti:

Percentuale dei Requisiti Obbligatori Soddisfatti (PROS)

È un valore percentuale che indica la quantità di requisiti obbligatori adempiti nel corso del progetto.

- Misurazione: $PROS = \frac{\text{requisiti obbligatori soddisfatti}}{\text{requisiti obbligatori totali}}$;
- Valore minimo accettabile: 100%;
- Valore preferibile: 100%.

2.2.1.2 Progettazione

Le metriche usate per l'attività di progettazione sono le seguenti:

Structural Fan-In (SFIN)

È un valore intero che indica quante componenti utilizzano un dato modulo. Uno *SFIN* alto sta a significare un consistente riuso della componente.

- Misurazione: valore intero che indica il conteggio delle componenti;
- Valore minimo accettabile: ≥ 0 ;
- Valore preferibile: ≥ 1 .

Structural Fan-Out (SFOUT)

È un valore intero che indica quante componenti vengono utilizzate dalla componente in questione. Uno *SFOUT* alto sta a significare un significativo accoppiamento della componente.

- Misurazione: valore intero che indica il conteggio delle componenti;
- Valore minimo accettabile: $= 0$;
- Valore preferibile: ≤ 6 .

2.2.1.3 Codifica

Le metriche usate per l'attività di codifica sono le seguenti:

Coupling Between Objects (CBO)

È un valore intero che indica il grado di accoppiamento tra le classi di oggetti. Una classe A si dice *accoppiata* ad una classe B se A usa metodi o variabili definite in B.

- Misurazione: valore intero;
- Valore minimo accettabile: $0 \leq \text{CBO} \leq 6$;
- Valore preferibile: $0 \leq \text{CBO} \leq 1$.

Complessità ciclomatica

È una metrica utilizzata per stimare la complessità di funzioni, moduli, metodi o classi di un programma. Ciò viene fatto mediante la determinazione del numero dei cammini linearmente indipendenti attraverso il grafo di controllo di flusso. Un valore troppo elevato indica un'eccessiva complessità del codice che rende conseguentemente l'attività di manutenzione più difficile. Allo stesso tempo, un valore eccessivamente basso potrebbe essere indicatore di una scarsa efficienza dei metodi.

- Misurazione: valore intero;
- Valore minimo accettabile: $1 \leq \text{complessità ciclomatica} \leq 15$;
- Valore preferibile: $1 \leq \text{complessità ciclomatica} \leq 10$.

Rapporto linee di codice per linee di commento

Indica il rapporto tra le righe di codice e le righe di commento ad esso corrispondenti, escludendo le righe vuote, al fine di stimare il livello di difficoltà di manutenibilità del codice. Un rapporto troppo basso, infatti, potrebbe essere indicatore di scarse informazioni necessarie alla comprensione del codice scritto.

- Misurazione: valore decimale;
- Valore minimo accettabile: $\frac{\text{linee di codice}}{\text{linee di commento}} \geq 0.25$;
- Valore preferibile: $\frac{\text{linee di codice}}{\text{linee di commento}} \geq 0.30$.

Livello di annidamento

È un valore intero che indica il livello di annidamento nei vari metodi tenendo conto della presenza di strutture di controllo annidate: un alto livello di annidamento risulta in codice complesso e di difficile manutenzione.

- Misurazione: valore intero;
- Valore minimo accettabile: $1 \leq \text{livello annidamento} \leq 7$;
- Valore preferibile: $1 \leq \text{livello annidamento} \leq 3$.

Numero di parametri per metodo

È un numero intero che indica il numero di parametri di un metodo. Un valore troppo elevato denota un grado di complessità eccessivamente elevato del metodo

- Misurazione: valore intero;
- Valore minimo accettabile: $0 \leq \text{numero totale attributi} \leq 8$;
- Valore preferibile: $0 \leq \text{numero totale attributi} \leq 4$.

Numero di attributi per classe

È un numero intero che indica il numero totale di attributi presenti all'interno di una classe. Un valore troppo elevato denota un carico di responsabilità eccessiva per la classe, suggerendo di scomporre la stessa seguendo principi quali il *Single Responsibility Principle*.

- Misurazione: valore intero;
- Valore minimo accettabile: $1 \leq \text{numero totale attributi} \leq 15$;
- Valore preferibile: $1 \leq \text{numero totale attributi} \leq 8$.

2.3 Processi di Supporto

2.3.0.1 Pianificazione

Le metriche usate per l'attività di pianificazione sono le seguenti:

Budget at Completion (BAC)

Equivale al budget totale allocato per il progetto.

- Misurazione: numero intero;
- Valore minimo accettabile: valore del preventivo con un errore massimo del 5%, ovvero $\text{preventivo}-5\% \leq BAC \leq \text{preventivo}+5\%$;
- Valore preferibile: pari al preventivo.

Cost Variance (CV)

Detta anche *Budget Variance*, in quanto misura il budget a disposizione nel corso di un progetto software. In particolare, la *Cost Variance* è la differenza tra *Earned Value* e *Actual Cost*, ovvero tra ciò che si aveva pianificato di spendere e ciò che si è effettivamente speso nel corso del progetto. Se la *Cost Variance* ha valore negativo significa che si è *over budget*, se è nulla si è *on budget*, mentre se è positiva si è *under budget*.

- Misurazione: $CV = EV - AC$;
- Valore minimo accettabile: 0;
- Valore preferibile: > 0 ;

Schedule Variance (SV)

La *Schedule Variance* è una misura che si usa per indicare lo stato di avanzamento di un progetto software e viene calcolata mediante la differenza tra *Earned Value* e *Planned Value*. Se la *Schedule Variance* ha valore negativo significa che lo stato di avanzamento del progetto è in ritardo rispetto alla pianificazione, se è nulla lo stato di avanzamento del progetto è nei tempi previsti, mentre se è positiva significa che si è in anticipo rispetto alla pianificazione.

- Misurazione: $SV = EV - PV$;
- Valore minimo accettabile: 0;
- Valore preferibile: > 0 .

Earned Value (EV)

Anche chiamata *Budgeted Cost of Work Performed (BCWP)* indica la quantità di lavoro compiuta al momento del calcolo. Viene calcolata dal budget di progetto come segue:

- Misurazione: $EV = \% \text{ di lavoro completato} \cdot BAC$;
- Valore minimo accettabile: ≥ 0 ;
- Valore preferibile: ≥ 0 ;

Actual Cost (AC)

Anche conosciuto come *Actual Cost of Work Performed (ACWP)* indica la quantità di budget spesa al momento del calcolo.

- Misurazione: numero intero;
- Valore minimo accettabile: $0 \leq AC < BAC$;
- Valore preferibile: $0 \leq AC < PV$.

Planned Value (PV)

Corrisponde al valore del lavoro pianificato al momento del calcolo, ovvero al denaro che si dovrebbe aver guadagnato fino a quel momento.

- Misurazione: $\% \text{ di lavoro pianificato} \cdot BAC$;
- Valore minimo accettabile: ≥ 0 ;
- Valore preferibile: ≥ 0 .

Correlazione tra CV e SV

Lo stato di un progetto è esprimibile dalla correlazione tra *Cost Variance* e *Schedule Variance*, in particolare:

1. **SV e CV positive:** il progetto è in anticipo rispetto alla pianificazione e rientra nel budget previsto;

2. **SV positiva, CV negativa:** il progetto è in anticipo rispetto alla pianificazione ma ha superato il budget allocato;
3. **SV negativa, CV positiva:** il progetto è in ritardo rispetto alla pianificazione ma rientra nel budget previsto;
4. **SV e CV negative:** il progetto è in ritardo rispetto alla pianificazione e ha superato il budget previsto.

2.3.0.2 Verifica

Le metriche usate per l'attività di verifica sono le seguenti:

Code Coverage

È la percentuale di linee di codice che sono state eseguite dai test dopo un'esecuzione.

- Misurazione: $CC = \frac{\text{linee di codice eseguite dal test}}{\text{linee di codice totali}}$;
- Valore minimo accettabile: 80%;
- Valore preferibile: 100%.

2.3.0.3 Documentazione

Le metriche usate per l'attività di documentazione sono le seguenti:

Indice di Gulpease

È un indice che valuta la leggibilità del testo. È tarato sulla lingua italiana e considera la lunghezza delle parole, il numero delle frasi ed il numero delle parole totali. Il valore risultante è compreso tra 0 e 100, dove un valore di indice più alto corrisponde ad un indice di leggibilità più semplice.

- Misurazione: $Gulpease = 89 + \frac{300 \cdot (\text{numero delle frasi}) - 10 \cdot (\text{numero delle lettere})}{\text{numero delle parole}}$;
- Valore minimo accettabile: ≥ 40 ;
- Valore preferibile: ≥ 60 .

Correttezza ortografica

Tutti i documenti devono essere privi di errori grammaticali od ortografici.

- Misurazione: numero intero che indica il numero di errori presenti nel testo;
- Valore minimo accettabile: 0;
- Valore preferibile: 0.

2.4 Processi Organizzativi

2.4.0.1 Gestione della Qualità

Le metriche usate per la gestione della qualità sono le seguenti:

Percentuale di metriche soddisfatte (PMS)

E' un valore percentuale che indica quante metriche raggiungono soglie accettabili sul totale delle metriche considerate. Un valore percentuale basso potrebbe essere indicatore di scarsa qualità, incorrettezza di calcolo o scelta metriche poco adeguate.

- $PMS = \frac{\text{numero di metriche soddisfatte}}{\text{numero totale di metriche}}$
- Valore minimo accettabile: 60%;
- Valore preferibile: 90%.

3 Qualità di Prodotto

3.1 Scopo

3.2 Funzionalità

3.3 Affidabilità

3.4 Usabilità

3.5 Manutenibilità

4 Test di Verifica

4.1 Test di Unità

Il test di unità ha come obiettivo quello di determinare la correttezza e la completezza, rispetto ai requisiti, di un programma visto come singolo modulo_G.

Questa tipologia di test verrà sviluppata in vista delle prossime revisioni.

4.2 Test di Integrazione

Il test di integrazione ha come obiettivo quello di verificare la correttezza funzionale nell'interazione tra più moduli_G. In particolare:

1. verifica sull'assemblamento dei vari moduli_G aggiunti incrementalmente;
2. verifica sull'assemblamento di tutti i moduli_G allo stesso tempo.

Questa tipologia di test verrà sviluppata in vista delle prossime revisioni.

4.3 Test di Sistema

Il test di sistema ha come obiettivo quello di testare particolari proprietà globali di esso. In particolare:

1. **test di stress**: valutazione del sistema in condizioni di sovraccarico;
2. **test di robustezza**: valutazione del sistema quando sono presenti dati non corretti;
3. **test di sicurezza**: valutazione del sistema nella sua sicurezza.

Questa tipologia di test verrà sviluppata in vista delle prossime revisioni.

4.4 Test di Regressione

Il test di regressione ha come obiettivo quello di verificare che ad ogni aggiornamento di un modulo_G software, la nuova versione mantenga le funzionalità di quella precedente. Il particolare si applica attraverso l'esecuzione della versione nuova e quella precedente sugli stessi dati, confrontando successivamente i risultati ottenuti.

Questa tipologia di test verrà sviluppata in vista delle prossime revisioni.

5 Test di Validazione

5.1 Test di Accettazione

In questa sezione sono riportati tutti i test definiti relativi ai requisiti che il prodotto finale dovrà superare.

In seguito sono riportati i seguenti test:

1. test funzionali;
2. test di qualità;
3. test di vincolo;
4. test prestazionali.

Relativamente ai requisiti prestazionali, non sono previsti test di accettazioni per quanto esposto nel documento *Analisi dei Requisiti*.

5.1.1 Test funzionali

Tabella 5.1.1: Tabella dei test funzionali

Test	Requisito e Descrizione	Implementato	Superato
TA2F1	R2F1: L'utente può leggere una breve guida iniziale riguardante l'applicativo e i comandi per effettuare l'accesso	No	No
TAR2F2	R2F2: L'utente può richiedere di visualizzare una descrizione più approfondita per ogni comando messo a disposizione da <i>Etherless-cli</i>	No	No
TA2F2.1	R2F2.1: Per ottenere informazioni specifiche su un comando, l'utente deve inserire il comando <i>help</i> seguito dal nome del comando di suo interesse	No	No
TA2F2.2	R2F2.2: Se il comando di cui si vogliono avere maggiori informazioni non è tra quelli messi a disposizione da <i>Etherless-cli</i> deve essere mostrato un messaggio di errore	No	No
TA1F3	R1F3: Un utente non registrato può richiedere la creazione di un nuovo account all'interno della rete Ethereum _G	No	No
TA1F3.1	R1F3.1: Una volta creato il nuovo account, il sistema deve mostrare nella CLI _G le credenziali a esso relative	No	No

Tabella 5.1.1: (continua)

Test	Requisito e Descrizione	Implementato	Superato
TA1F3.1.1	R1F3.1.1: A seguito del completamento della procedura di registrazione viene mostrato l'address associato al nuovo account creato	No	No
TA1F3.1.2	R1F3.1.2: A seguito del completamento della procedura di registrazione viene mostrata la private key associata al nuovo account creato	No	No
TA2F3.1.3	R2F3.1.3: A seguito del completamento della procedura di registrazione viene mostrata la mnemonic phrase associata al nuovo account creato	No	No
TA2F3.2	R2F3.2: L'utente può richiedere il salvataggio su file delle credenziali dell'account creato durante la procedura di registrazione	No	No
TA1F4	R1F4: Un utente può effettuare il login	No	No
TA1F4.1	R1F4.1: Un utente si può autenticare manualmente tramite l'utilizzo del comando <i>login</i>	No	No
TA1F4.1.1	R1F4.1.1: Per completare la procedura di login manuale l'utente deve inserire il proprio address	No	No
TA1F4.1.2	R1F4.1.2: Per completare la procedura di login manuale l'utente deve inserire la propria private key	No	No
TA2F4.1.3	R2F4.1.3: L'utente può decidere di completare la procedura di login manuale utilizzando la propria mnemonic phrase al posto della private key	No	No
TA2F4.2	R2F4.2: Durante la procedura di login manuale l'utente può richiedere che le proprie credenziali siano memorizzate per accessi futuri	No	No
TA2F4.3	R2F4.3: L'utente si può autenticare tramite login automatico	No	No
TA1F5	R1F5: L'utente può effettuare il logout	No	No
TA2F6	R2F6: L'utente può richiedere di visualizzare l'address associato alla sessione corrente	No	No

Tabella 5.1.1: (continua)

Test	Requisito e Descrizione	Implementato	Superato
TA1F7	R1F7: L'utente può richiedere di visualizzare la descrizione dettagliata di una funzione tramite il comando <i>info</i>	No	No
TA1F7.1	R1F7.1: Per visualizzare la descrizione di una funzione l'utente deve inserire il nome della funzione di interesse	No	No
TA1F7.2	R1F7.2: Nel caso in cui l'utente richieda di visualizzare la descrizione di una funzione non presente nel sistema, deve essere mostrato un messaggio di errore	No	No
TA2F8	R2F8: Il sistema deve permettere all'utente di cercare una funzione attraverso una keyword	No	No
TA2F8.1	R2F8.1: Per effettuare la ricerca è necessario che l'utente inserisca una keyword	No	No
TA2F8.2	R2F8.2: A seguito di una ricerca il sistema deve mostrare la lista di tutte le funzioni che presentano la keyword indicata all'interno del proprio nome	No	No
TA2F8.2.1	R2F8.2.1: La visualizzazione di un risultato di ricerca include la firma della funzione	No	No
TA2F8.2.2	R2F8.2.2: La visualizzazione di un risultato di ricerca include il costo di esecuzione della funzione	No	No
TA2F8.2.3	R2F8.2.3: La visualizzazione di un risultato di ricerca include l'address del creatore della funzione	No	No
TA2F8.3	R2F8.3: Se una ricerca non porta a nessun risultato deve essere mostrato un messaggio di errore	No	No
TA1F9	R1F9: L'utente deve essere in grado di eseguire le funzioni messe a disposizione da <i>Etherless</i> attraverso il comando <i>run</i>	No	No
TA1F9.1	R1F9.1: Per eseguire una funzione è necessario inserire il relativo nome	No	No

Tabella 5.1.1: (continua)

Test	Requisito e Descrizione	Implementato	Superato
TA1F9.1.1	R1F9.1.1: Nel caso in cui il nome inserito a seguito del comando <i>run</i> non corrisponda ad alcuna funzione presente nel sistema, deve essere visualizzato un messaggio di errore	No	No
TA1F9.2	R1F9.2: L'esecuzione di una funzione necessita dell'inserimento dei parametri necessari per la sua esecuzione	No	No
TA1F9.2.1	R1F9.2.1: Se l'utente tenta di eseguire una funzione inserendo un numero di parametri che non coincide con quanto richiesto, deve essere visualizzato un messaggio di errore	No	No
TA1F9.3	R1F9.3: A seguito dell'esecuzione di una funzione il sistema deve mostrare all'utente i relativi risultati	No	No
TA1F9.4	R1F9.4: Nel caso in cui l'utente richieda di eseguire una funzione senza avere credito sufficiente, deve essere mostrato un messaggio di errore	No	No
TA1F10	R1F10: L'utente deve essere in grado di visualizzare tutte le funzioni disponibili in <i>Etherless</i> tramite il comando <i>list</i>	No	No
TA2F10.1	R2F10.1: L'utente può richiedere di visualizzare solo le funzioni da lui caricate tramite l'utilizzo di un apposito flag	No	No
TA1F10.2	R1F10.2: La visualizzazione di un elemento della lista ottenuta a seguito del comando <i>list</i> include la firma della funzione	No	No
TA1F10.3	R1F10.3: La visualizzazione di un elemento della lista ottenuta a seguito del comando <i>list</i> include il costo di esecuzione della funzione	No	No
TA1F10.4	R1F10.4: La visualizzazione di un elemento della lista ottenuta a seguito del comando <i>list</i> include il creatore della funzione	No	No
TA1F10.5	R1F10.5: Nel caso in cui il risultato del comando <i>list</i> sia vuoto, deve essere visualizzato un apposito messaggio	No	No

Tabella 5.1.1: (continua)

Test	Requisito e Descrizione	Implementato	Superato
TA1F11	R1F11: L'utente deve essere in grado di eseguire il deploy_G di una propria funzione all'interno della piattaforma <i>Etherless</i>	No	No
TA1F11.1	R1F11.1: Per eseguire il deploy_G l'utente deve inserire il percorso del file contenente il codice della funzione	No	No
TA2F11.1.	R2F11.1.1: Se il formato del file indicato durante la procedura di deploy_G non è supportato dall'applicativo deve essere mostrato un messaggio di errore	No	No
TA1F11.1.	R1F11.1.2: Se il file indicato durante la procedura di deploy_G non esiste, deve essere visualizzato un messaggio di errore	No	No
TA1F11.2	R1F11.2: Per eseguire il deploy_G l'utente deve inserire il nome della funzione considerata	No	No
TA1F11.2.	R1F11.2.1: Nel caso in cui il nome della funzione di cui si tenta di fare il deploy_G sia troppo lungo, deve essere visualizzato un messaggio di errore	No	No
TA1F11.2.	R1F11.2.2: Nel caso in cui il nome della funzione di cui si tenta di fare il deploy_G sia già usato nel sistema, deve essere visualizzato un messaggio di errore	No	No
TA2F11.3	R2F11.3: Per eseguire il deploy_G l'utente deve inserire una descrizione della funzione	No	No
TA2F11.3.	R2F11.3.1: Se la descrizione inserita durante la procedura di deploy_G supera la lunghezza massima, deve essere mostrato un messaggio di errore	No	No
TA1F11.4	R1F11.4: Nel caso in cui l'utente tenti di eseguire il deploy_G di una funzione senza avere il credito necessario, deve essere visualizzato un messaggio di errore	No	No
TA1F12	R1F12: L'utente deve essere in grado di modificare le informazioni relative ad una funzione da lui caricata	No	No

Tabella 5.1.1: (continua)

Test	Requisito e Descrizione	Implementato	Superato
TA1F12.1	R1F12.1: Per eseguire la procedura di modifica è necessario che l'utente indichi il nome della funzione che vuole modificare	No	No
TA1F12.1.1	R1F12.1.1: Nel caso in cui, durante la procedura di modifica, l'utente inserisca il nome di una funzione non presente all'interno della piattaforma <i>Etherless</i> , deve essere mostrato un messaggio di errore	No	No
TA1F12.1.2	R1F12.1.2: Nel caso in cui, durante la procedura di modifica, l'utente inserisca il nome di una funzione che non è di sua proprietà, deve essere mostrato un messaggio di errore	No	No
TA1F12.2	R1F12.2: Il sistema deve permettere all'utente di modificare la descrizione associata ad una propria funzione	No	No
TA1F12.2.1	R1F12.2.1: L'utente deve visualizzare un errore nel caso in cui, durante la procedura di modifica, venga inserita una descrizione di lunghezza superiore a quella massima consentita	No	No
TA1F12.3	R1F12.3: Il sistema deve permettere all'utente di aggiornare il codice di una propria funzione	No	No
TA1F12.3.1	R1F12.3.1: Se il file indicato durante la procedura di aggiornamento del codice di una funzione non esiste, deve essere mostrato un messaggio di errore	No	No
TA2F13	R2F13: L'utente deve essere in grado di visualizzare la propria cronologia di richieste di esecuzione	No	No
TA2F13.1	R2F13.1: L'utente deve poter essere in grado di richiedere di visualizzare solo una porzione della propria cronologia di esecuzione	No	No
TA2F13.2	R2F13.2: La visualizzazione di un elemento della cronologia include l'identificativo della richiesta di esecuzione	No	No
TA2F13.3	R2F13.3: La visualizzazione di un elemento della cronologia include il nome della funzione richiesta	No	No

Tabella 5.1.1: (continua)

Test	Requisito e Descrizione	Implementato	Superato
TA2F13.4	R2F13.4: La visualizzazione di un elemento della cronologia include il valore dei parametri indicati nella chiamata alla funzione	No	No
TA2F13.5	R2F13.5: La visualizzazione di un elemento della cronologia include il risultato della richiesta di esecuzione	No	No
TA2F13.6	R2F13.6: La visualizzazione di un elemento della cronologia include la data e orario della richiesta	No	No
TA1F14	R1F14: L'utente deve essere in grado di eliminare una funzione da lui caricata	No	No
TA1F14.1	R1F14.1: Per eseguire l'operazione di eliminazione l'utente deve inserire il nome della funzione da eliminare	No	No
TA1F14.1.1	R1F14.1.1: Nel caso in cui il nome inserito durante la procedura di eliminazione non si riferisca ad alcuna funzione presente all'interno del sistema, deve essere mostrato un messaggio di errore	No	No
TA1F14.1.2	R1F14.1.2: Nel caso in cui la funzione considerata nella procedura di eliminazione non sia di proprietà dell'utente, deve essere visualizzato un messaggio di errore	No	No

5.1.2 Test di qualità

Tabella 5.1.2: Tabella dei test di qualità

Test	Requisito e Descrizione	Implementato	Superato
TA1Q1	R1Q1: La progettazione e la codifica devono rispettare le norme e le metriche definite nei documenti <i>Norme di Progetto v1.0.0</i> e <i>Piano di Qualifica v1.0.0</i>	No	No
TA1Q2	R1Q2: Il sistema deve essere pubblicato con licenza MIT	No	No

Tabella 5.1.2: (continua)

Test	Requisito e Descrizione	Implementato	Superato
TA1Q3	R1Q3: Il codice sorgente di <i>Etherless</i> deve essere pubblicato e versionato usando Github _G o GitLab _G	No	No
TA1Q4	R1Q4: Deve essere redatto un manuale sviluppatore	No	No
TA1Q4.1	R1Q4.1: Il manuale sviluppatore deve contenere le informazioni per eseguire e fare il deploy _G dei moduli _G	No	No
TA1Q5	R1Q5: Deve essere redatto un manuale utente	No	No
TA1Q5.1	R1Q5.1: Il manuale utente deve contenere tutte le informazioni necessarie all'utente finale per utilizzare correttamente il sistema	No	No
TA1Q6	R1Q6: La documentazione per l'utilizzo del software deve essere scritta in lingua inglese.	No	No
TA1Q7	R1Q7: Nella scrittura del codice JavaScript _G deve essere seguita la guida sullo stile di programmazione Airbnb JavaScript style guide	No	No
TA1Q8	R1Q8: Lo sviluppo del codice JavaScript _G deve essere supportato dal software di analisi statica del codice ESLint _G	No	No

5.1.3 Test di vincolo

Tabella 5.1.3: Tabella dei test di qualità

Test	Requisito e Descrizione	Implementato	Superato
TA1V1	R1V1: Gli smart contract _G devono essere scritti in Solidity _G	No	No
TA1V2	R1V2: Gli smart contract _G devono poter essere aggiornati	No	No
TA1V3	R1V3: L'applicativo deve essere sviluppato utilizzando TypeScript 3.6 _G	No	No
TA1V3.1	R1V3.1: Deve essere utilizzato il meccanismo delle promise/async-await _G come approccio principale	No	No

Tabella 5.1.3: (continua)

Test	Requisito e Descrizione	Implementato	Superato
TA1V4	R1V4: Il modulo _G <i>Etherless-server</i> deve essere implementato utilizzando il Framework Serverless _G	No	No
TA1V5	R1V5: Il progetto deve utilizzare i seguenti ambienti di sviluppo: ambiente di sviluppo locale, ambiente di testing e ambiente di staging	No	No
TA2V5.1	R2V5.1: Gli ambienti per la fase di sviluppo locale e testing possono fare utilizzo della rete TestRPC fornita dal framework Truffle	No	No
TA2V5.2	R2V5.2: Per la fase di staging è desiderabile l'utilizzo della rete Ethereum _G Ropsten _G	No	No
TA1V5.3	R1V5.3: Durante la fase di staging l'applicativo deve essere pubblicamente accessibile	No	No
TA1V5.4	R1V5.4: Al termine del progetto il prodotto deve essere pronto per la produzione	No	No
TA3V5.4.1	R3V5.4.1: L'ambiente di produzione deve fare utilizzo dell'Ethereum main network	No	No
TA3V6	R3V6: Il pagamento deve essere gestito tramite un meccanismo di escrow	No	No
TA1V7	R1V7: Deve essere possibile installare <i>Etherless-cli</i> usando npm (node package manager)	No	No

A Standard di qualità

A.1 ISO/IEC 9126

ISO/IEC 9126 è lo standard generale usato per valutare la qualità del software. Esso è articolato in quattro parti:

1. Modello per la qualità del software, che a sua volta è suddiviso in:
 - Modello per la qualità esterna ed interna;
 - Modello per la qualità in uso.
2. Metriche per la qualità interna
3. Metriche per la qualità esterna
4. Metriche per la qualità in uso

A.1.1 Modello per la qualità del software

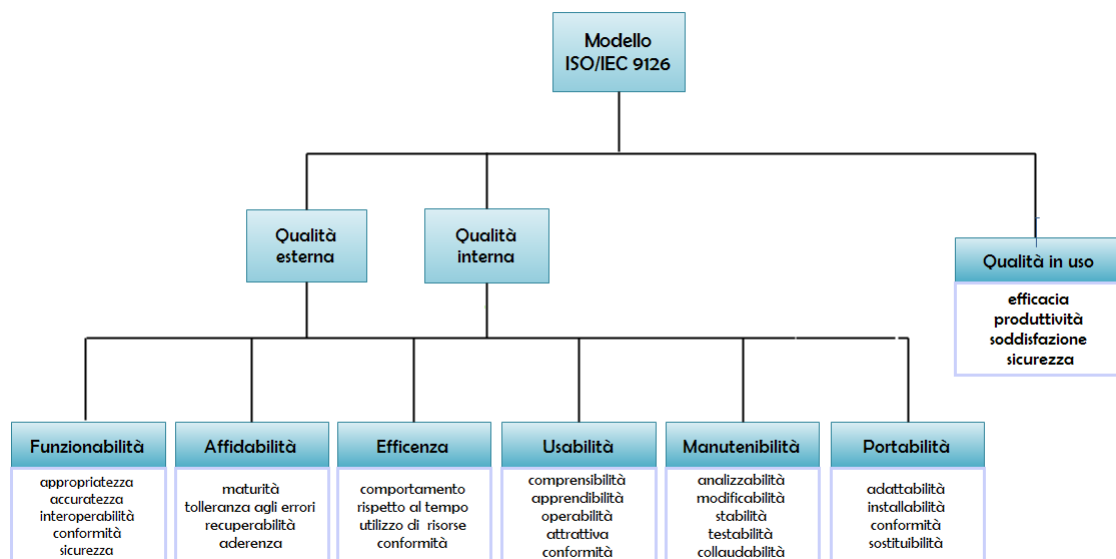


Figura A.1.1: Modello ISO/IEC 9126 (fonte: Wikipedia)

A.1.1.1 Modello per la qualità esterna ed interna

Il modello per la definizione della qualità interna ed esterna è composto da sei caratteristiche generali e varie sotto caratteristiche misurabili attraverso delle metriche. Tali caratteristiche sono:

Funzionalità

E' la capacità del software di fornire funzioni che soddisfano esigenze stabilite nell'Analisi dei Requisiti e che permettono di operare in condizioni specifiche. Questa capacità si traduce nelle seguenti sotto caratteristiche:

- **Appropriatezza:** capacità di fornire funzioni appropriate per attività specifiche che permettano di raggiungere gli obiettivi prefissati;
- **Accuratezza:** capacità di fornire risultati corretti e con la precisione richiesta;
- **Interoperabilità:** capacità di interagire con uno o più sistemi specificati;
- **Conformità:** capacità di aderire a standard rilevanti al settore in esame;
- **Sicurezza:** capacità di proteggere informazioni e dati.

Affidabilità

E' la capacità del software di mantenere uno dato livello di prestazioni quando usato in specifiche condizioni. Questa capacità si traduce nelle seguenti sotto caratteristiche:

- **Maturità:** capacità di evitare il verificarsi di errori o malfunzionamenti in fase di esecuzione;
- **Tolleranza agli errori:** capacità di mantenere livelli predeterminati di prestazioni anche in presenza di malfunzionamenti o errori;
- **Recuperabilità:** capacità di ripristinare il livello di prestazioni e di recupero delle informazioni rilevanti in seguito ad un malfunzionamento;
- **Aderenza:** capacità di aderire a standard e regole inerenti all'affidabilità.

Efficienza

E' la capacità del software di eseguire le funzioni prefissate minimizzando il tempo necessario e sfruttando al meglio le risorse disponibili. Questa capacità si traduce nelle seguenti sotto caratteristiche:

- **Nel tempo:** capacità di fornire appropriati tempi di risposta;
- **Nello spazio:** capacità di utilizzare un appropriato numero di risorse.

Usabilità

E' la capacità del software di essere capito, appreso, usato ed accettato positivamente dall'utente. Questa capacità si traduce nelle seguenti sotto caratteristiche:

- **Comprensibilità:** capacità di essere chiaro riguardo le proprie funzionalità ed il proprio utilizzo;
- **Apprendibilità:** capacità di essere facilmente apprendibile;
- **Operabilità:** capacità di permettere all'utente di raggiungere i suoi scopi e controllarne l'uso;
- **Attrattività:** capacità di essere piacevole per l'utente che ne fa uso;
- **Conformità:** capacità di aderire a standard o convenzioni relativi all'usabilità.

Manutenibilità

E' la capacità del software di essere modificato includendo correzioni, miglioramenti od adattamenti. Questa capacità si traduce nelle seguenti sotto caratteristiche:

- **Analizzabilità:** capacità di essere facilmente analizzato al fine di individuare un errore;
- **Modificabilità:** capacità di 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 al fine di validare le modifiche apportate.

Portabilità

E' la capacità del software di essere trasportato da un ambiente hardware/software ad un altro seguendo le evoluzioni tecnologiche. Questa capacità si traduce nelle seguenti sotto caratteristiche:

- **Adattabilità:** capacità di essere facilmente adattato a differenti ambienti operativi senza applicare modifiche;
- **Installabilità:** capacità di essere installato in uno specifico ambiente;
- **Conformità:** capacità di aderire a standard e convenzioni relative alla portabilità;
- **Sostituibilità:** capacità di essere utilizzato al posto di un altro software per svolgere gli stessi compiti nello stesso ambiente.

A.1.1.2 Modello per la qualità in uso

Il modello per la definizione della qualità in uso elenca quattro caratteristiche generali che permettono agli utenti di ottenere specifici obiettivi. Tali caratteristiche sono:

- **Efficacia:** capacità del software di permettere agli utenti di raggiungere gli obiettivi specificati con accuratezza e completezza;
- **Produttività:** capacità del software di essere efficiente rispetto alle risorse necessarie;
- **Soddisfazione:** capacità del software di soddisfare gli utenti;
- **Sicurezza:** capacità del software di avere dei livelli di rischio accettabili rispetto a danni nei confronti di persone, apparecchiature e ambiente operativo.

A.1.2 Metriche per la qualità interna

La qualità interna viene rilevata tramite analisi statica, ovvero le metriche scelte vengono applicate a software non eseguibile e permettono di individuare eventuali problemi che potrebbero influire sulla qualità finale del prodotto. Le misure effettuate permettono di prevedere il livello di qualità esterna ed in uso del prodotto finale.

A.1.3 Metriche per la qualità esterna

La qualità esterna viene rilevata tramite analisi dinamica. Le metriche sono applicate al software in esecuzione e ne misurano il comportamento attraverso attività di test in funzione degli obiettivi prefissati. Idealmente la qualità esterna determina la qualità in uso.

A.1.4 Metrica per la qualità in uso

Si tratta di metriche applicabili solo al prodotto finito ed in uso in condizioni reali. La qualità in uso viene raggiunta solo se è stato raggiunto sia il livello di qualità interna che di qualità esterna.

A.2 ISO/IEC 15504**A.3 Ciclo di Deming**

B Valutazioni per il miglioramento

Questa sezione riporta i problemi riscontrati dal gruppo *Roundabout* durante il corso del progetto. Ogni problema viene valutato per trovare una possibile soluzione e quindi un miglioramento il più efficace ed efficiente possibile.

Si espongono di seguito i problemi incontrati divisi in 3 raggruppamenti:

- **organizzazione:** problemi relativi all'organizzazione e la comunicazione all'interno del gruppo;
- **ruoli:** problemi relativi allo svolgimento dei diversi ruoli;
- **strumenti di lavoro:** problemi relativi l'uso degli strumenti utilizzati.

B.1 Valutazioni sull'organizzazione

Tabella B.1.1: Valutazioni Organizzazione

Problema	Soluzione
Riunioni Interne: si è rivelato un problema organizzativo l'impossibilità di vedersi fisicamente a causa della situazione di emergenza COVID-19 _G	Abbiamo concordato di utilizzare maggiormente strumenti di collaborazione che consentono, oltre alla possibilità di effettuare videochiamate, una comunicazione semplificata per i diversi problemi che si possono verificare.
Appuntamenti: Problema a definire una calendarizzazione degli incontri tra i vari membri del gruppo	Abbiamo definito che le riunioni interne saranno effettuate cadenzialmente due volte alla settimana il martedì e il venerdì, salvo esigenze particolari.
Riunioni Esterne: Durante la prima riunione effettuata con il <i>Proponente_G</i> a mezzo Skype _G , si è valutato il problema comune di connessione instabile e conseguente perdita di parole durante la conversazione.	Risolto proponendo al <i>Proponente_G</i> incontri telematici su piattaforma Zoom _G , molto più leggera e con limitati problemi di chiamata.

B.2 Valutazioni sui ruoli

Tabella B.2.1: Valutazioni Ruoli

Problema	Soluzione
Rivestire un ruolo: Il problema comune a tutti i ruoli è stato quello di doversi adattare ad una mentalità diversa in base al contesto richiesto, considerato il vincolo che ogni membro dovrà ricoprire un ruolo descritto nelle <i>Norme di Progetto</i> .	Valutato che il maggior impatto di questa problematica si verifica nella fase iniziale di ogni cambio di ruolo, si è deciso di limitare le rotazioni indicativamente ogni due settimane cercando di non lasciare lavori in sospeso al membro successivo. In ogni caso vige il buon senso e la collaborazione reciproca.
Responsabile di Progetto:	
Amministratore:	
Analista:	
Verificatore:	

B.3 Valutazioni sugli strumenti di lavoro

Tabella B.3.1: Valutazioni Strumenti di Lavoro

Problema	Soluzione
L^AT_EX: si è rivelato un problema l'utilizzo di questo strumento, in quanto la maggior parte del gruppo <i>Roundabout</i> non lo aveva mai utilizzato prima.	La soluzione è stata quella di usufruire dell'esperienza maturata da parte di alcuni membri del gruppo per apprendere le basi di utilizzo: prima creando un template standard, poi illustrandolo assieme ad alcuni comandi che avremmo utilizzato con maggiore frequenza.
Ethereum_G: si è rivelato un problema la non conoscenza di questa piattaforma	Si è colmata questa mancanza tramite ricerca personale e studio autonomo.
Omogeneità dei documenti prodotti: Considerato che la stesura di un documento può essere effettuata anche da più persone che ricoprono lo stesso ruolo in contemporanea, si è verificato il problema di omogeneità all'interno dei documenti	La soluzione migliore è stata quella di concordare assieme nelle <i>Norme di Progetto</i> gli utilizzi di maiuscole, minuscole, corsivo, grassetto, etc.

C Resoconto delle attività di verifica

C.1 Verifiche statiche

Ogni documento prodotto è stato analizzato da parte dei *Verificatori*, adottando un metodo Walkthrough_G ed Inspection_G.

Terminata questa analisi, in accordo con il redattore, si procede alla risoluzione di lacune eventualmente presenti.

C.2 Verifiche requisiti

Questo tipo di verifica è necessario per accertarsi che, la relazione tra casi d'uso, requisiti e fonti non abbia discrepanze. Per facilitare questa verifica si è scelto di utilizzare il software PragmaDB.

C.3 Verifiche automatizzate

Nella seguente tabella vengono riportati i valori di Gulpease_G calcolati per ogni documento. I calcoli sono stati effettuati escludendo le intestazioni e le note a piè di pagina, in modo da avere un risultato valido ed attendibile. L'esito della verifica è da intendersi *Positivo* qualora l'indice di Gulpease abbia valore maggiore di 40.

Tabella C.3.1: Verifica Gulpease documenti

Documento	Gulpease	Esito
Analisi dei Requisiti v1.0.0	0	Non Positivo
Glossario v1.0.0	0	Non Positivo
Norme di Progetto v1.0.0	0	Non Positivo
Piano di Progetto v1.0.0	0	Non Positivo
Studio di Fattibilità v1.0.0	0	Non Positivo
Verbale xyz v1.0.0	0	Non Positivo
Verbale xyz v1.0.0	0	Non Positivo
Verbale xyz v1.0.0	0	Non Positivo
Verbale xyz v1.0.0	0	Non Positivo
Verbale xyz v1.0.0	0	Non Positivo