



<CodeHex16>

unipd.codehex16@gmail.com

Piano di Qualifica

Data 03/01/2024

Versione 0.4.0

Sommario

Piano di qualifica

Ruoli

Matteo Bazzan	Redattore
Luca Ribon	Verificatore
Francesco Fragonas	
Gabriele Magnelli	Redattore
Filippo Sabbadin	Redattore
Luca Rossi	
Yi Hao Zhuo	Verificatore

Registro delle Versioni

Versione	Data	Autore	Cambiamenti	Verificatore
0.4.0	04/03/2025	Gabriele Magnelli	Aggiunta sezione cruscotto di valutazione delle metriche	Luca Ribon
0.3.0	27/02/2025	Filippo Sabbadin	Aggiunta sezioni di valutazione	Luca Ribon
0.2.0	05/02/2025	Matteo Bazzan	Aggiunta metriche di qualità	Luca Ribon
0.1.0	08/01/2024	Filippo Sabbadin	Prima stesura	Yi Hao Zhuo

Indice

1. Introduzione e scopo	1
1.1. Scopo del documento	1
1.2. Glossario	1
1.3. Versioni e maturità	1
1.4. Riferimenti	1
1.4.1. Riferimenti normativi	1
1.4.2. Riferimenti informativi	1
2. Metriche di qualità	3
2.1. Qualità di processo	3
2.1.1. Fornitura	3
2.1.2. Sviluppo	4
2.1.3. Documentazione	4
2.1.4. Verifica	5
2.1.5. Gestione della qualità	6
2.2. Qualità del prodotto	6
2.2.1. Funzionalità	6
2.2.2. Affidabilità	7
2.2.3. Usabilità	8
2.2.4. Efficienza	9
2.2.5. Manutenibilità	9
2.2.6. Sicurezza	10
3. Metodologie di testing	11
3.1. Tipologie di test	11
3.1.1. Test di Unità	11
3.1.2. Test di Integrazione	12
3.1.3. Test di Sistema	12
3.1.4. Test di Accettazione	13
4. Cruscotto di valutazione delle metriche	14
4.1. MPC-EAC(Estimated At Completion)	14
4.2. MPC-EV(Estimated Value) - MPC-PV(Planned Value)	15
4.3. MPC-AC(Actual Cost) - MPC-ETC(Estimated To Completion)	16
4.4. MPC-SC(Schedule Variance) - MPC-CV(Cost Variance)	17
4.5. MPC-RSI(Requirements Stability Index)	18
4.6. MPC-Correttezza Ortografica	19

4.7. MPC-Indice Gulpease	20
4.8. MPC-Non-Calculated-Risk	21
4.9. MPC-QMS(Quality Metric Satisfied)	22
5. Processi di automiglioramento	23
5.1. Introduzione	23
5.2. Valutazione sull'organizzazione, pianificazione e incontri	23
5.3. Valutazione sui ruoli	24
5.4. Valutazione sugli strumenti	25

Lista di immagini

Figura 1	Stima del costo totale durante i vari sprint*	14
Figura 2	Stima dei valori di PV e EV durante i vari sprint.	15
Figura 3	Stima dei valori di AC e ETC durante i vari sprint.	16
Figura 4	Stima dei valori di CV e SV durante i vari sprint.	17
Figura 5	Stima di RSI durante i vari sprint.	18
Figura 6	Stima dei valori di correttezza ortografica durante i vari sprint.	19
Figura 7	Stima dei valori dell'indice Gulpease per ogni documento durante i vari sprint.	20
Figura 8	Stima dei valori di CV e SV durante i vari sprint.	21
Figura 9	Stima della percentuale di metriche di qualità soddisfatte durante i vari sprint.	22

Lista di tabelle

Tabella 1	Valori per misurare la qualità della fornitura	4
Tabella 2	Valori per misurare la qualità dello sviluppo	4
Tabella 3	Valori per misurare la qualità della documentazione	5
Tabella 4	Valori per misurare la qualità del processo di verifica	6
Tabella 5	Valori per misurare la gestione della qualità	6
Tabella 6	Valori per misurare la qualità del prodotto in termini di funzionalità	6
Tabella 7	Valori per misurare la qualità del prodotto in termini di affidabilità .	8
Tabella 8	Valori per misurare la qualità del prodotto in termini di usabilità . . .	8
Tabella 9	Valori per misurare la qualità del prodotto in termini di efficienza . .	9
Tabella 10	Valori per misurare la qualità del prodotto in termini di manutenibilità	10
Tabella 11	Valori per misurare la qualità del prodotto in termini di sicurezza ..	10
Tabella 12	Tipologie di test	11
Tabella 13	Esempi di unit test	12
Tabella 14	Esempi di test di integrazione	12
Tabella 15	Esempi di test di sistema	13
Tabella 16	Esempi di test di accettazione	13
Tabella 17	Valutazione sull'organizzazione	23
Tabella 18	Valutazione sui ruoli	24
Tabella 19	Valutazione sugli strumenti utilizzati	25

Lista di equazioni

Equazione (1)	5
Equazione (2)	7

1. Introduzione e scopo

1.1. Scopo del documento

In questo documento vengono dichiarate tutte le metriche che il gruppo [CodeHex16*](#) userà per misurare la [qualità*](#) del prodotto e dei processi usati per la realizzazione del progetto.

1.2. Glossario

Per facilitare la comprensione di questo documento, viene fornito un glossario che chiarisce il significato dei termini specifici utilizzati nel contesto del progetto. Ogni termine di glossario è contrassegnato con un asterisco «*» in apice e collegato direttamente alla pagina web del glossario, permettendo così di accedere immediatamente alla definizione completa del termine.

Le definizioni sono disponibili nel documento `Glossario.pdf` e nella seguente pagina web: <https://codehex16.github.io/glossario.html>

1.3. Versioni e maturità

Data la natura evolutiva del documento, questa versione potrebbe non rappresentare la versione finale. Il documento continuerà a subire modifiche per garantire una maggiore correttezza e chiarezza nel testo per facilitare la comprensione e lettura.

1.4. Riferimenti

1.4.1. Riferimenti normativi

- Capitolo C7 - [LLM*](#) : [Assistente Virtuale*](#)

<https://www.math.unipd.it/~tullio/IS-1/2024/Progetto/C7.pdf>

- [Norme di Progetto*](#)

1.4.2. Riferimenti informativi

- Slide T08 - Qualità di processo

<https://codehex16.github.io/resources/slides/T8.pdf>

- Slide T09 - Verifica e validazione: introduzione

<https://codehex16.github.io/resources/slides/T9.pdf>

- Verifica e validazione:
 - ISO/IEC 9126:2001 SWE Product Quality;
 - ISO/IEC 14598:1999 SW Product Evaluation;
 - ISO/IEC 25000:2005 SQuaRE: Systems and software Quality;
 - ISO/IEC 25010:2011 Quality model;
 - ISO/IEC 25020:2019 Quality measurement framework;
 - ISO/IEC 25030:2007 Quality requirements;
 - ISO/IEC 25040:2011 Quality evaluation;
 - ISO 9000:2015 (fondamenti e glossario);
 - ISO 9001:2015 (sistema qualità - requisiti);
 - ISO/IEC/IEEE 90003:2018 (versione applicata ai prodotti SW)
 - ISO 9004:2018 (qualità organizzativa - autovalutazione)
 - ISO/IEC 33020:2019.

2. Metriche di qualità

2.1. Qualità di processo

2.1.1. Fornitura

Per il processo di fornitura vengono indicate tutte le scelte operative fatte in fase di sviluppo. L'acronimo usato prima dei nomi è [MPC*](#): Minimum Predictive Capability. Questa metrica viene usata in Machine Learning per misurare la capacità di un modello di generare previsioni precise. Nel nostro caso, l'MPC è il valore minimo da raggiungere per essere considerato accettabile.

- **MPC-CC - Completion Cost:** costo finale raggiunto alla fine del progetto. Idealmente non deve superare quello stimato durante le fasi iniziali;
- **MPC-EC - Estimated Cost:** costo stimato calcolando le ore necessarie per lo sviluppo del progetto;
- **MPC-BAC - [Budget*](#) At Completion:** costo totale del progetto preventivato per il suo completamento;
- **MPC-AC - Actual Cost:** budget speso/utilizzato fino a quel determinato momento;
- **MPC-ETC - Estimated To Completion:** stima del costo finale alla data della misurazione;
- **MPC-EV - Earned Value:** valore ottenuto fino a quel dato momento, si basa sui progressi del completamento delle attività. Il valore viene calcolato tramite il prodotto di BAC per la percentuale del lavoro attualmente svolto;
- **MPC-PV - Planned Value:** attività lavorativa decisa, cioè da completare, entro la data prevista. Si basa sulla programmazione delle attività del progetto e riflette il valore del lavoro che si intende portare a termine. Questo valore è calcolato tramite il prodotto di BAC per la percentuale del lavoro che deve essere completato (rispetto all'intero progetto) entro la data presa in considerazione;
- **MPC-CV - Cost Variance:** valore che misura la differenza tra il budget disponibile e il quello usato effettivamente fino a quel momento. Il valore viene calcolato come differenza tra Ev e AC;
- **MPC-SV - Schedule Variance:** varianza rispetto a quanto previsto inteso come anticipo o ritardo sui tempi delle attività svolte e da svolgere. Questo valore viene calcolato come la differenza tra EV e PV;
- **MPC-EAC - Estimated At Completion:** valore stimato per i compiti da svolgere. Questo valore viene dato dalla divisione di BAC per CPI(Cost Performance Index).

Metrica	Nome	Valore accettabile	Valore ottimo
MPC-CC	Completion Cost	$\leq 105\%$ EC	$\leq 100\%$ EC
MPC-AC	Actual Cost	≥ 0	\leq EAC
MPC-ETC	Estimated To Completion	$\geq 0\%$	\leq EAC
MPC-EV	Earned Value	≥ 0	\leq EAC
MPC-PV	Planned Value	≥ 0	\leq BAC
MPC-CV	Cost Variance	$\geq -5\%$	≥ 0
MPC-SV	Schedule Variance	$\geq -10\%$	≥ 0
MPC-EAC	Estimated At Completion	$\pm 5\%$ BAC	$=$ BAC

Tabella 1: Valori per misurare la qualità della fornitura

2.1.2. Sviluppo

- **MPC-RSI - Requirements Stability Index:** indice di stabilità dei requisiti. Indica la percentuale di requisiti che sono stati modificati rispetto al totale dei requisiti. Un valore alto indica che i requisiti sono stabili e non soggetti a modifiche frequenti.
- **MPC-TD - Technical Debt Ratio:** rapporto tra il tempo necessario per risolvere i problemi tecnici e il tempo necessario per sviluppare nuove funzionalità. Un valore basso indica che il codice è ben strutturato e non presenta problemi tecnici.

Metrica	Nome	Valore accettabile	Valore ottimo
MPC-RSI	Requirements Stability Index	$\geq 80\%$	100%
MPC-TD	Technical Debt Ratio	$\leq 15\%$	$\leq 5\%$

Tabella 2: Valori per misurare la qualità dello sviluppo

2.1.3. Documentazione

- **MPC-IG - Indice di Gulpease**

Indica la complessità nella lettura di una frase o documento. Considera come variabili il numero di parole, di frasi e di lettere.

Formula dell'indice di Gulpease:

$$89 + \frac{(300 * \text{numero di frasi}) - (10 * \text{numero di lettere})}{\text{numero di parole}} \quad (1)$$

• MPC-CO - Correttezza ortografica

Indica il numero di errori ortografici presenti nella documentazione.

Metrica	Nome	Valore accettabile	Valore ottimo
MPC-IG	Indice di Gulpease	≥40	≥60
MPC-CO	Correttezza ortografica	3	0

Tabella 3: Valori per misurare la qualità della documentazione

2.1.4. Verifica

• MPC-CCO - Code coverage

Quantità di codice eseguito durante i test.

Viene utilizzato per valutare la qualità dei test e garantire che il codice sia stato adeguatamente testato. Un alto livello indica che il codice è stato eseguito in molti contesti e scenari diversi con diverse parti di codice. In altre parole, indica quanto codice è stato sottoposto ai test.

• MPC-TSP - Test superati in percentuale

Indica la proporzione di test automatizzati o manuali che sono stati eseguiti con successo rispetto al totale dei test previsti. Viene espressa come una percentuale e serve a misurare quanto dell'applicazione in fase di sviluppo è stato verificato con successo tramite i test. Una percentuale alta di test superati indica che il sistema è stabile e che la maggior parte delle funzionalità funzionano come previsto.

In altre parole, indica quanti test sono stati superati.

Metrica	Nome	Valore accettabile	Valore ottimo
MPC-CCO	Code coverage	≥90%	100%
MPC-TSP	Test superati in percentuale	100%	100%

Tabella 4: Valori per misurare la qualità del processo di verifica

2.1.5. Gestione della qualità

- **MPC-SQM - Satisfaction of Quality Metrics:** misura la quantità di metriche soddisfatte. Questo valore viene calcolato come la somma delle metriche di qualità soddisfatte diviso il numero totale di metriche di qualità.

Metrica	Nome	Valore accettabile	Valore ottimo
MPC-SQM	Satisfaction of Quality Metrics	≥85%	100%

Tabella 5: Valori per misurare la gestione della qualità

2.2. Qualità del prodotto

2.2.1. Funzionalità

- **MPD-RO - Copertura requisiti obbligatori:** indica la percentuale di requisiti obbligatori coperti dal prodotto. Un valore del 100% indica che tutti i requisiti obbligatori sono stati implementati.
- **MPD-OP - Copertura requisiti opzionali:** indica la percentuale di requisiti opzionali coperti dal prodotto. Un valore del 100% indica che tutti i requisiti opzionali sono stati implementati.

Metrica	Nome	Valore accettabile	Valore ottimo
MPD-RO	Copertura requisiti obbligatori	100%	100%
MPD-OP	Copertura requisiti opzionali	≥50%	100%

Tabella 6: Valori per misurare la qualità del prodotto in termini di funzionalità

2.2.2. Affidabilità

- **MPD-CC - Code coverage:** indica la percentuale di codice coperto dai test. Un valore alto indica che il codice è stato testato in modo approfondito e che è meno probabile che contenga errori.
- **MPD-BC - [Branch](#)* coverage:** sottoinsieme di code coverage, indica la percentuale di rami delle condizioni che sono stati eseguiti durante i test. Un valore alto indica che il codice è stato testato in modo approfondito e che è meno probabile che contenga errori.
- **MPD-SC - Statement coverage:** sottoinsieme di code coverage, indica la percentuale di istruzioni che sono state eseguite durante i test. Un valore alto indica che il codice è stato testato in modo approfondito e che è meno probabile che contenga errori. Viene calcolato con la seguente formula:

$$\frac{\text{numero di istruzioni eseguite}}{\text{numero di istruzioni totali nel codice}} * 100 \quad (2)$$

- **MPD-FT - Failure Tolerance:** indica la capacità del prodotto di mantenere un livello di prestazioni accettabile anche in caso di guasti o malfunzionamenti. Un valore alto indica che il prodotto è in grado di gestire i guasti senza compromettere le funzionalità principali.
- **MPD-FF - Failure Frequency:** indica la frequenza con cui si verificano guasti o malfunzionamenti nel prodotto. Un valore basso indica che il prodotto è affidabile e presenta pochi problemi.
- **MPD-MTBF - Mean Time Between Failures:** indica il tempo medio tra un guasto e il successivo. Un valore alto indica che il prodotto è affidabile e presenta pochi guasti.
- **MPD-DS - Disponibilità sistema:** indica la percentuale di tempo in cui il sistema è operativo. Un valore alto indica che il sistema è affidabile e che è disponibile per l'utente.

Metrica	Nome	Valore accettabile	Valore ottimo
MPD-CC	Code coverage	$\geq 80\%$	100%
MPD-BC	Branch coverage	$\geq 50\%$	$\geq 80\%$
MPD-SC	Statement coverage	$\geq 60\%$	$\geq 80\%$
MPD-FT	Failure Tolerance	100%	100%
MPD-FF	Failure Frequency	0	0
MPD-MTBF	Mean Time Between Failures	$\geq 48h$	$\geq 72h$
MPD-DS	Disponibilità sistema	$\geq 90\%$	$\geq 99.9\%$

Tabella 7: Valori per misurare la qualità del prodotto in termini di affidabilità

2.2.3. Usabilità

- **MPD-TA - Tempo di apprendimento:** indica il tempo necessario per un utente base per apprendere come utilizzare il prodotto. Un valore basso indica che il prodotto è facile da usare e richiede poco tempo per essere appreso. Viene calcolato con sessioni di test con utenti.
- **MPD-EUA - Errori utente/azione:** indica il numero di errori commessi dagli utenti durante l'utilizzo del prodotto. Un valore basso indica che il prodotto è intuitivo e facile da usare. Viene calcolato tramite log delle interazioni.
- **MPD-TSR - [Task*](#) success rate:** indica la percentuale di task completati con successo dagli utenti. Un valore alto indica che il prodotto è facile da usare e che gli utenti riescono a completare le azioni richieste. Viene calcolato con sessioni di test con utenti.

Metrica	Nome	Valore accettabile	Valore ottimo
MPD-TA	Tempo di apprendimento	≤ 15 min (utente base)	≤ 5 min
MPD-EUA	Errori utente/azione	≤ 0.5 errori/azione	0
MPD-TSR	Task success rate	$\geq 75\%$	100%

Tabella 8: Valori per misurare la qualità del prodotto in termini di usabilità

2.2.4. Efficienza

- **MPD-TRA - Tempo risposta API***: tempo di risposta delle API per il 90% delle richieste. Un valore basso indica che il sistema risponde velocemente alle richieste degli utenti.
- **MPD-MP - Memoria processo**: indica l'utilizzo della memoria da parte del sistema. Un valore basso indica che il sistema utilizza in modo efficiente le risorse disponibili.
- **MPD-CE - Consumo energetico**: indica il consumo energetico del sistema. Un valore basso indica che il sistema consuma poca energia.

Metrica	Nome	Valore accettabile	Valore ottimo
MPD-TRA	Tempo risposta API	≤500 ms	≤200 ms
MPD-MP	Memoria processo	≤512 MB	≤256 MB
MPD-CE	Consumo energetico	≤2% batteria/min	≤1% batteria/min

Tabella 9: Valori per misurare la qualità del prodotto in termini di efficienza

2.2.5. Manutenibilità

- **MPD-CCL - Complessità ciclomatica**: misura la complessità del codice. Un valore basso indica che il codice è semplice e facile da mantenere.
- **MPD-DT - Debito Tecnico**: indica la percentuale di debito tecnico rispetto al codice totale. Un valore basso indica che il codice è ben strutturato e non presenta problemi tecnici.
- **MPD-CSD - Code Smell density**: indica il numero di «code smells» (cattive pratiche di codifica) per 100 righe di codice. Un valore basso indica che il codice è ben strutturato e non presenta problemi tecnici.
- **MPD-TFB - Tempo fix bug**: tempo medio per risolvere un bug critico. Un valore basso indica che il team è in grado di risolvere i bug in modo rapido ed efficiente.

Metrica	Nome	Valore accettabile	Valore ottimo
MPD-CCL	Complessità ciclomatica	≤ 15 per modulo	≤ 10
MPD-DT	Debito Tecnico	$\leq 15\%$	$\leq 5\%$
MPD-CSD	Code Smell density	≤ 5 smell/100 righe	0 smell
MPD-TFB	Tempo fix bug	≤ 4 ore (critico)	≤ 2 ore

Tabella 10: Valori per misurare la qualità del prodotto in termini di manutenibilità

2.2.6. Sicurezza

- **MPD-AF - Tasso di autenticazione fallita:** percentuale di tentativi di autenticazione falliti. Un valore basso indica che il sistema è sicuro e che è difficile per gli utenti non autorizzati accedere al sistema.
- **MPD-CRD - Crittografia dati:** livello di crittografia dei dati sensibili. Un valore alto indica che i dati sono protetti e che è difficile per gli utenti non autorizzati accedere ai dati sensibili.

Metrica	Nome	Valore accettabile	Valore ottimo
MPD-AF	Tasso di autenticazione fallita	$\leq 5\%$	$\leq 1\%$
MPD-CRD	Crittografia dati	100% dati sensibili	100% dati sensibili

Tabella 11: Valori per misurare la qualità del prodotto in termini di sicurezza

3. Metodologie di testing

3.1. Tipologie di test

Per garantire la qualità del prodotto, il team CodeHex16 ha deciso di verificare il prodotto attraverso diverse tipologie di test.

Tipo di test	Scopo	Strumenti
Test di Unità	Verificare il corretto funzionamento delle singole componenti	pytest, vitest
Test di Integrazione	Validare l'interazione tra moduli e servizi	Postman
Test di Sistema	Verificare il comportamento end-to-end rispetto ai requisiti funzionali	Playwright
Test di Accettazione	Validare il sistema con il #gloss[committente]/utente finale	Checklist manuali

Tabella 12: Tipologie di test

Ad ogni test è stato attribuito un codice univoco per identificarlo strutturato nel seguente modo: [TIP0]-XXX, dove XXX è un numero progressivo.

- Test di Unità: TU-XXX
- Test di Integrazione: TI-XXX
- Test di Sistema: TS-XXX
- Test di Accettazione: TA-XXX

Per ogni test viene specificato lo **stato** di completamento, che può essere:

- Superato
- Pianificato
- Fallito
- Non implementato

3.1.1. Test di Unità

I test di unità sono utilizzati per verificare il corretto funzionamento delle singole componenti del software. Vengono scritti dai programmatori e sono eseguiti in modo automatico. Gli strumenti utilizzati per i test di unità sono pytest e vitest.

Esempi applicati al progetto:

Codice	Descrizione	Stato
TU-001	Verifica del parsing delle credenziali durante il login	Superato
TU-002	Test generazione risposta LLM con input validi/invalidi	Non implementato
TU-003	Controllo formati logo supportati (PNG, JPG, SVG)	Superato

Tabella 13: Esempi di unit test

3.1.2. Test di Integrazione

I test di integrazione sono utilizzati per validare l'interazione tra moduli e servizi. Vengono scritti dai programmatori e sono eseguiti in modo automatico. Lo strumento utilizzato per i test di integrazione è Postman.

Esempi applicati al progetto:

Codice	Descrizione	Stato
TI-001	Integrazione modulo autenticazione con database utenti	Superato
TI-002	Comunicazione tra frontend e API di generazione risposte LLM	Superato
TI-003	Verifica sincronizzazione impostazioni tema (dark/light mode) su più dispositivi	Non Implementato

Tabella 14: Esempi di test di integrazione

3.1.3. Test di Sistema

I test di sistema sono utilizzati per verificare il comportamento end-to-end del sistema rispetto ai requisiti funzionali. Vengono scritti dagli analisti e sono eseguiti in modo automatico. Lo strumento utilizzato per i test di sistema è Playwright.

Esempi applicati al progetto:

Codice	Descrizione	Stato
TS-001	Test completo flusso cliente: Login → Richiesta prodotto → Valutazione risposta	Non Implementato
TS-002	Test gestione fornitori: Aggiunta #gloss[account] → Configurazione #gloss[chatbot] → Caricamento documenti	Superato
TS-003	Test tolleranza ai fallimenti: Simulazione downtime sistema durante l'invio messaggi	Pianificato

Tabella 15: Esempi di test di sistema

3.1.4. Test di Accettazione

I test di accettazione sono utilizzati per validare il sistema con il committente o l'utente finale. Vengono scritti dagli analisti e sono eseguiti in modo manuale. Gli strumenti utilizzati per i test di accettazione sono checklist manuali.

Esempi applicati al progetto:

Codice	Descrizione	Stato
TA-001	Verifica personalizzazione interfaccia	Non Implementato
TA-002	Validazione accuratezza risposte LLM	Non Implementato
TA-003	Test flusso cliente: Registrazione → Richiesta prodotto → Valutazione risposta	Non Implementato

Tabella 16: Esempi di test di accettazione

4. Cruscotto di valutazione delle metriche

4.1. MPC-EAC(Estimated At Completion)

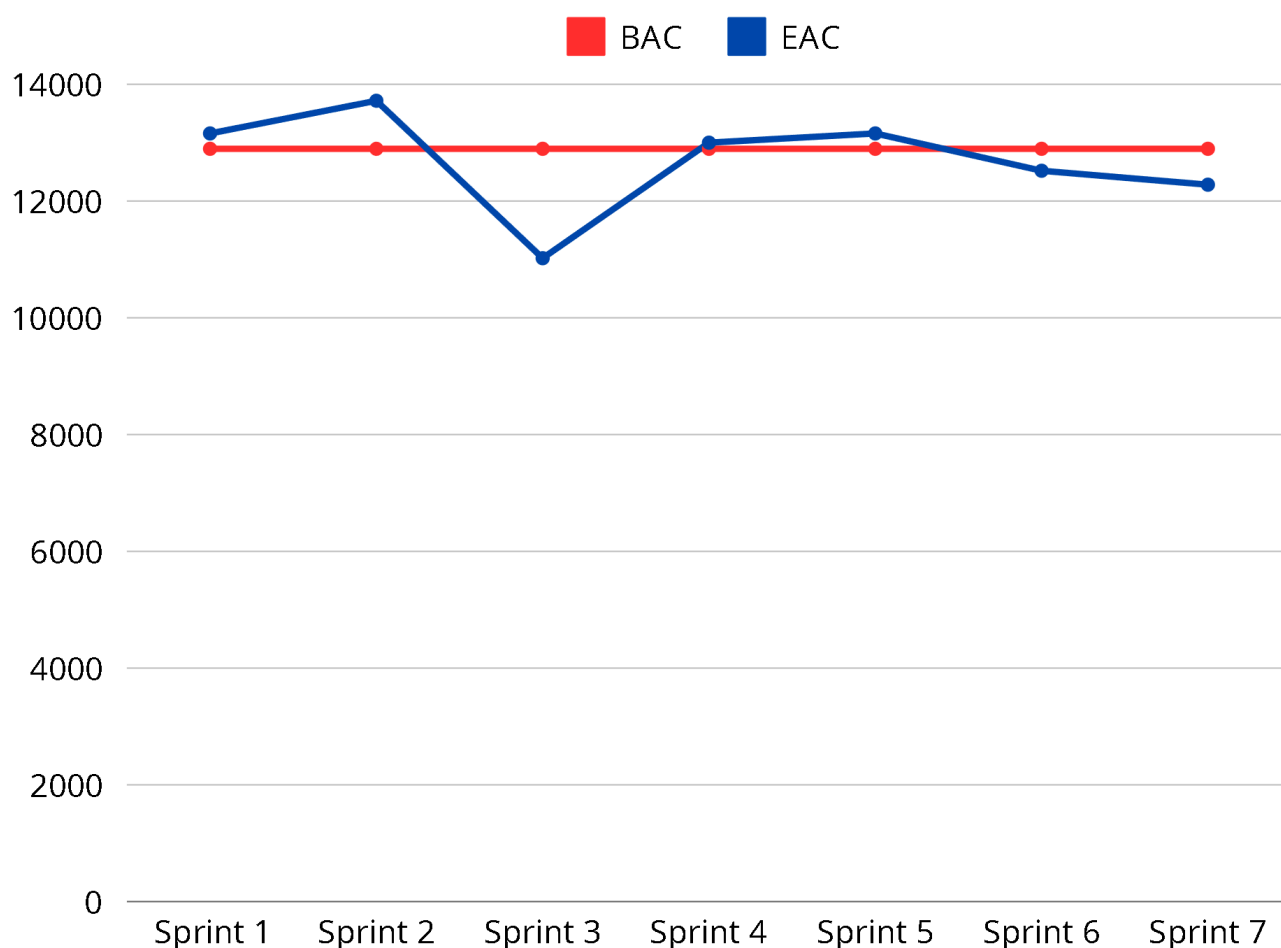


Figura 1: Stima del costo totale durante i vari [sprint*](#).

Osservando il grafico si nota come, soprattutto all'inizio, le stime dei costi totali si discostavano dal costo preventivato (BAC). Ciò è dovuto, per lo più, al fatto che all'inizio avevamo previsto un maggior numero di ore per il ruolo di Analista al fine di redigere il documento [Analisi dei Requisiti*](#) negli sprint 2 e 3 per questo, in tali periodi, si è verificata una diminuzione dei costi. Invece i rialzi sono più contenuti e sono dovuti, principalmente, ai periodi in cui avevamo bisogno di ore aggiuntive per il ruolo di verificatore.

4.2. MPC-EV(Estimated Value) - MPC-PV(Planned Value)

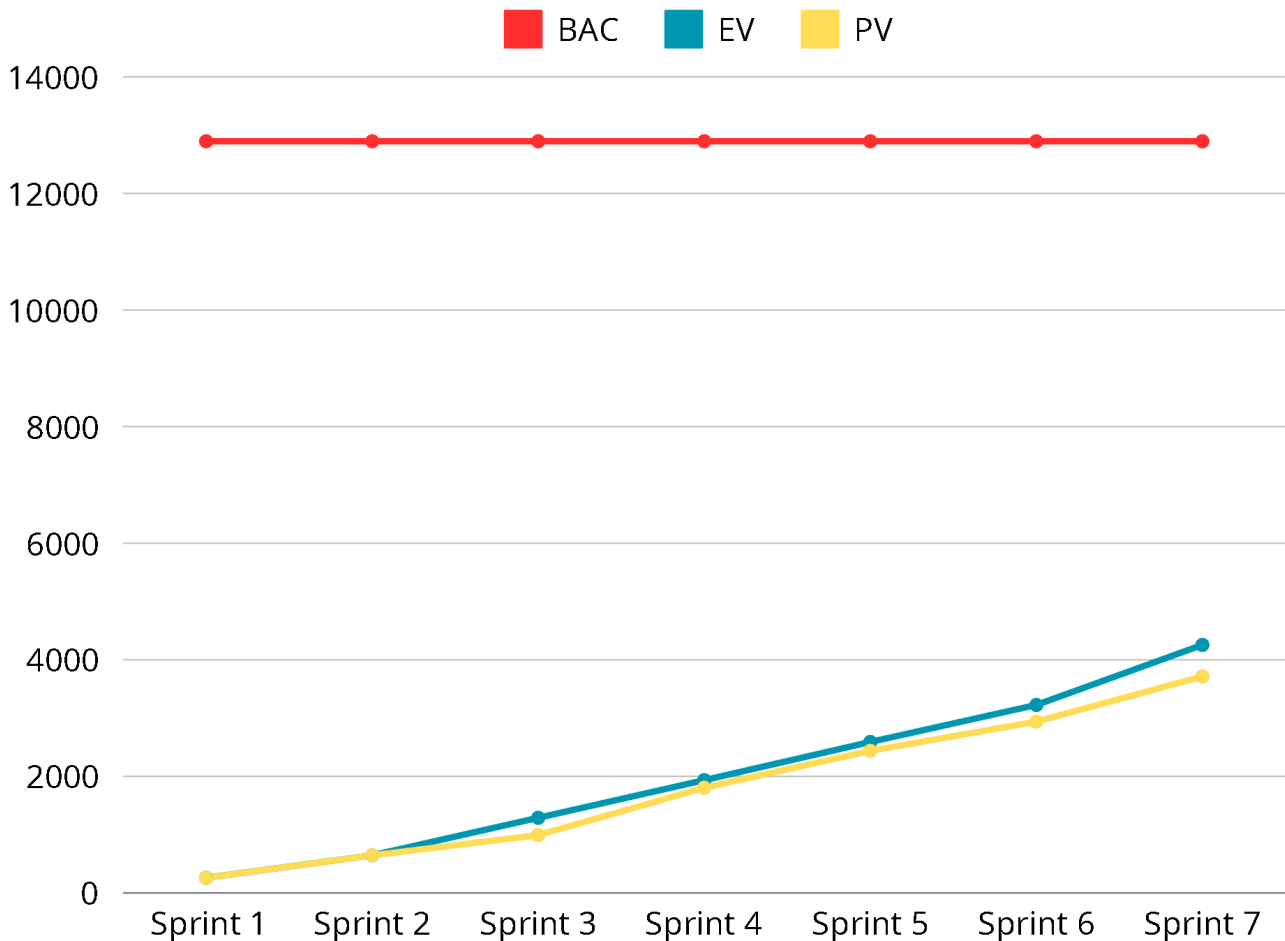


Figura 2: Stima dei valori di PV e EV durante i vari sprint.

Dal grafico si può notare come i valori di PV e EV quasi si sovrappongano e ciò è, dovuto al fatto che il lavoro effettivamente svolto è conforme a quello pianificato anche se con qualche piccola eccezione, per esempio, tra gli sprint 6 e 7 in cui vari componenti del team hanno avuto degli esami da svolgere e il lavoro e la produttività è leggermente diminuita.

4.3. MPC-AC(Actual Cost) - MPC-ETC(Estimated To Completion)

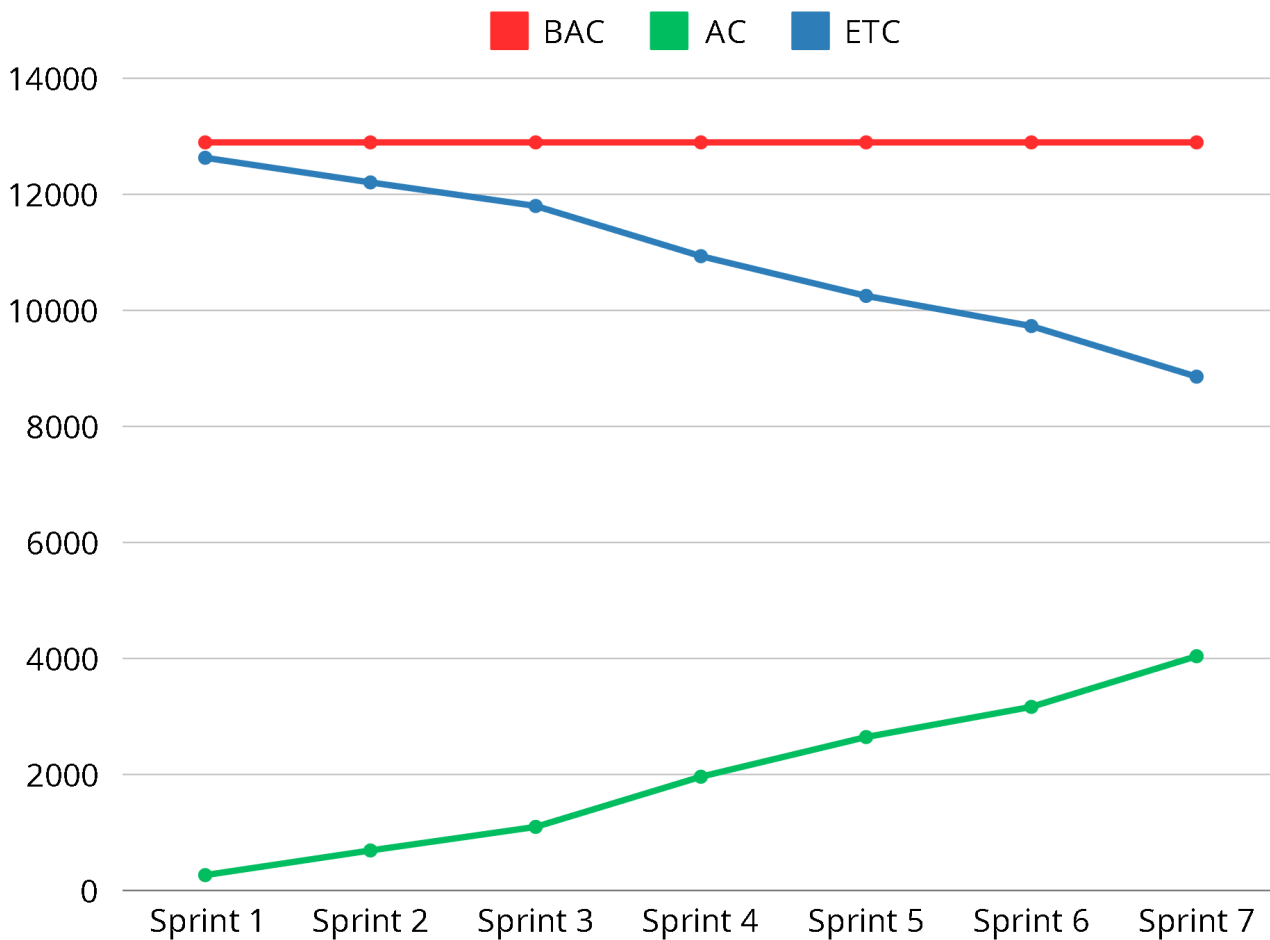


Figura 3: Stima dei valori di AC e ETC durante i vari sprint.

Il grafico mostra i valori di ETC(Estimated To Completion), cioè la stima del budget rimanente al team per portare al termine il progetto durante i vari sprint. Il grafico mostra anche i valori di AC(Actual Cost), cioè il budget effettivamente speso dal team durante i vari periodi. Si nota che l'ETC, giustamente, diminuisce costantemente con l'avanzare degli sprint, mentre l'AC rispetta una crescita proporzionale alla velocità con cui l'ETC decresce.

4.4. MPC-SC(Schedule Variance) - MPC-CV(Cost Variance)

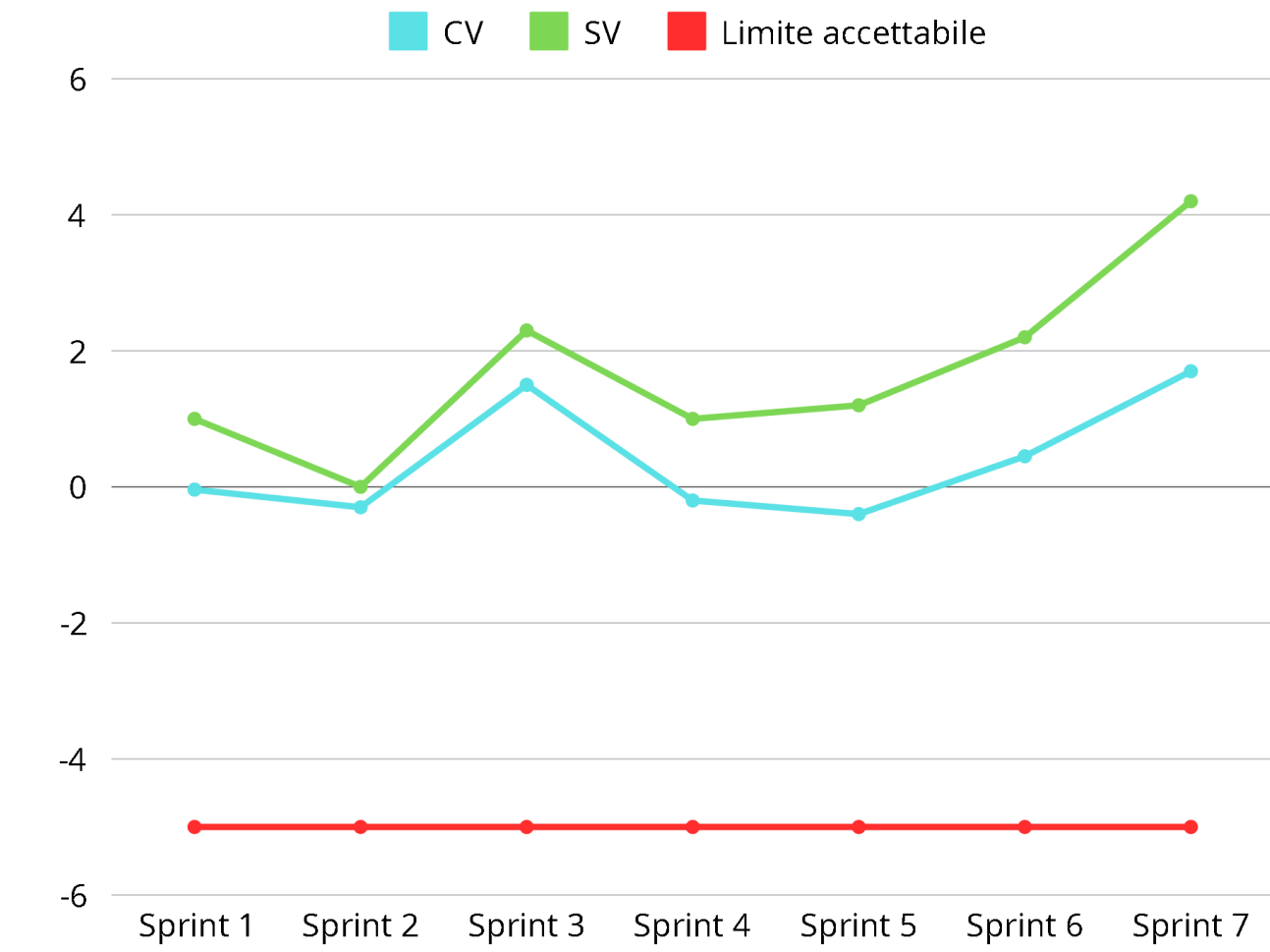


Figura 4: Stima dei valori di CV e SV durante i vari sprint.

Il grafico mostra i valori di SV(Schedule Variance) dati dalla differenza tra il valore guadagnato (EV) e il valore pianificato (PV) in percentuale e i valori di CV(Cost Variance) dati dalla differenza tra il valore ottenuto (EV) e il budget speso (AC) in percentuale. In generale la CV si avvicina in vari periodi allo 0 segno dove vi è una corrispondenza tra i costi sostenuti e l'avanzamento nel progetto, solo in alcuni casi il team è riuscito a fare un po' di lavoro in più rispetto a quello preventivato (probabilmente il team aveva preventivato al ribasso). Anche i valori di SV sono vicini allo 0 o sopra di qualche punto e quindi anche in questo caso sono stati rispettati i tempi previsti, solo in alcuni casi il gruppo aveva preventivato di usare più tempo per determinate attività al fine di procedere più velocemente con il progetto. Entrambi i valori comunque si discostano

al massimo di 4 punti percentuali SV e al massimo di 2 punti percentuali per i valori di CV.

4.5. MPC-RSI(Requirements Stability Index)

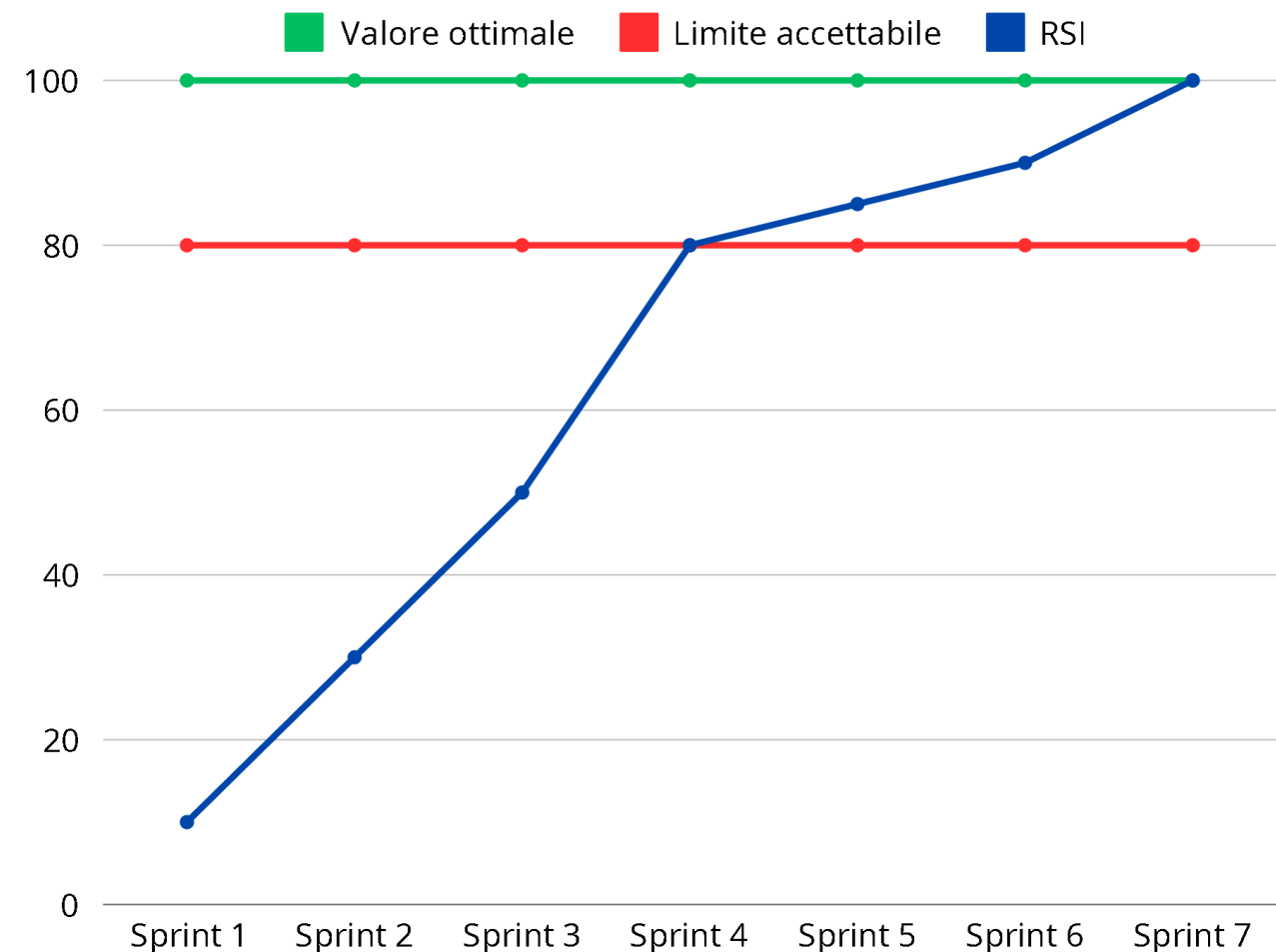


Figura 5: Stima di RSI durante i vari sprint.

Il grafico mostra l'RSI(Requirements Stability Index), usata per valutare la stabilità dei requisiti del progetto nel corso del tempo. Si nota che tra gli sprint 1 e 4 vi è una rapida crescita, infatti questi sono i periodi principali in cui il team ha usato il ruolo dell'analista al fine di redigere il documento Analisi dei Requisiti. Nei periodi successivi i requisiti sono stati migliorati e non vi sono state modifiche sostanziali ai requisiti fino allo sprint 7, in cui l'indice RSI risulta essere del 100%.

4.6. MPC-Correttezza Ortografica

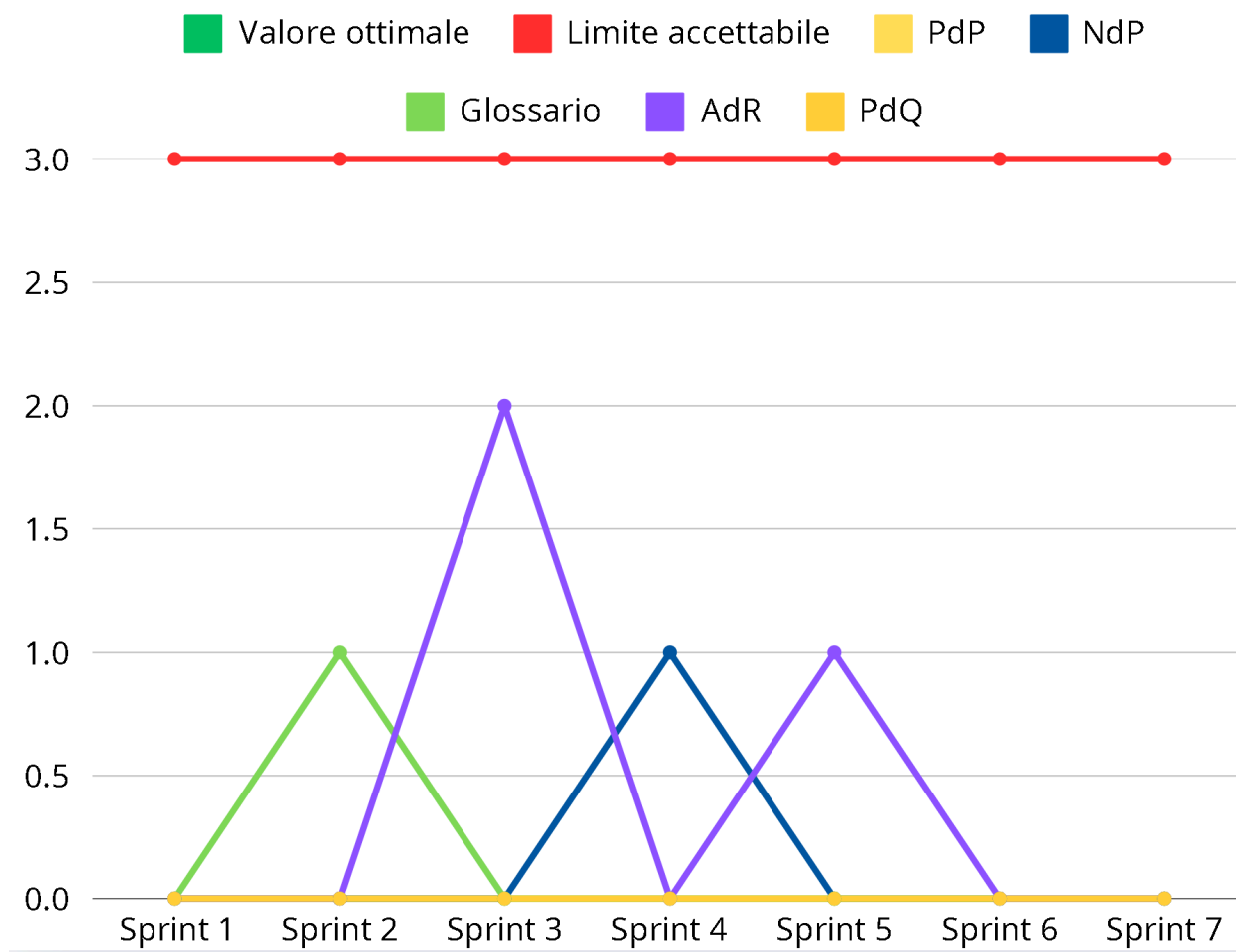


Figura 6: Stima dei valori di correttezza ortografica durante i vari sprint.

Dal grafico emerge che, nella maggior parte dei documenti, gli errori ortografici sono molto limitati, purtroppo qualche errore è sfuggito, ma è stato successivamente corretto. Nonostante ciò per buona parte dei documenti e del tempo gli errori presenti erano 0, in particolare si è raggiunto un ottimo risultato negli ultimi sprint.

4.7. MPC-Indice Gulpease

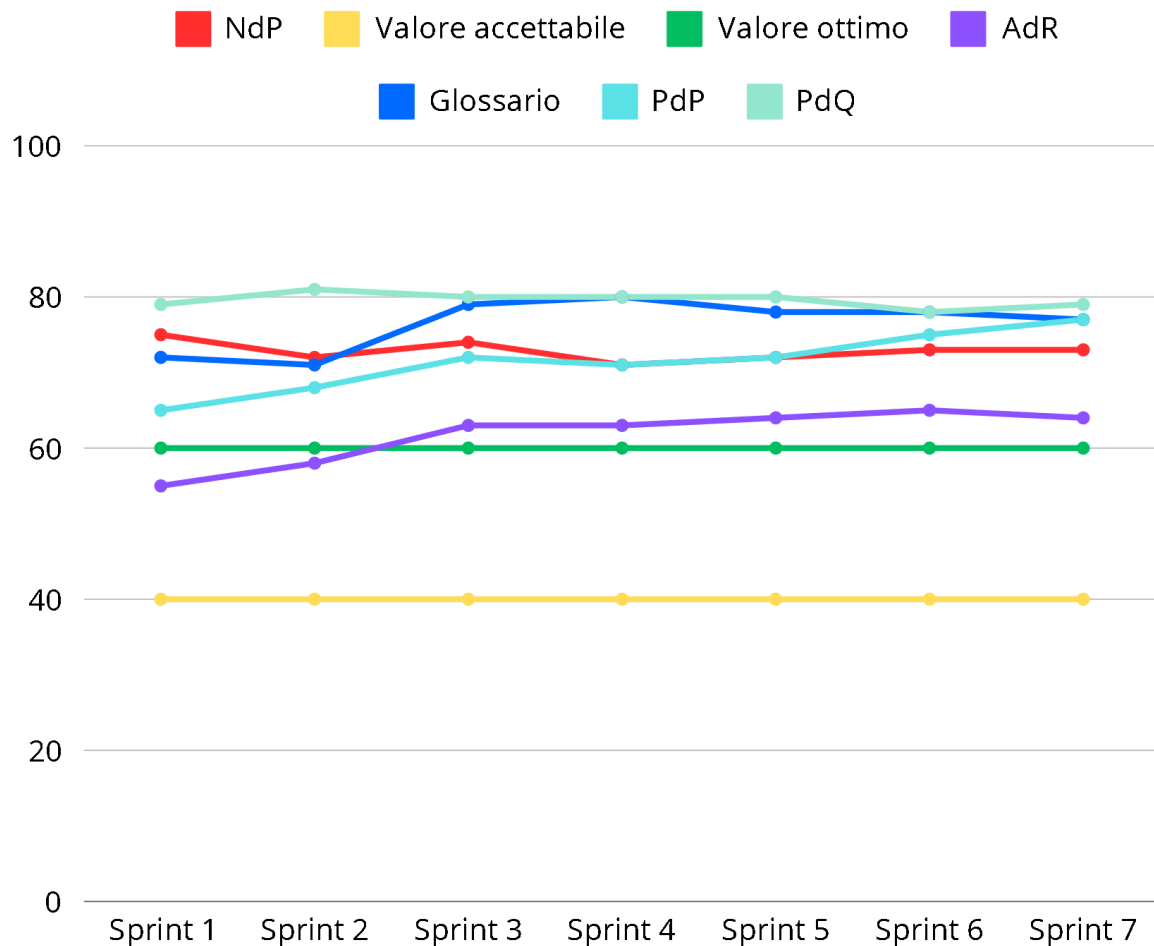


Figura 7: Stima dei valori dell'indice Gulpease per ogni documento durante i vari sprint.

Dal grafico si può osservare come, per la maggior parte dei documenti nei vari sprint vi sia stato un aumento, o una stabilizzazione dell'indice Gulpease. L'unico documento che inizia al di sotto del limite ottimo è l'Analisi dei Requisiti e questo è causato dalla specificità degli argomenti trattati e dal linguaggio utilizzato. Per il resto dei documenti l'indice Gulpease è sopra l'ottimo per la maggior parte degli sprint.

4.8. MPC-Non-Calculated-Risk

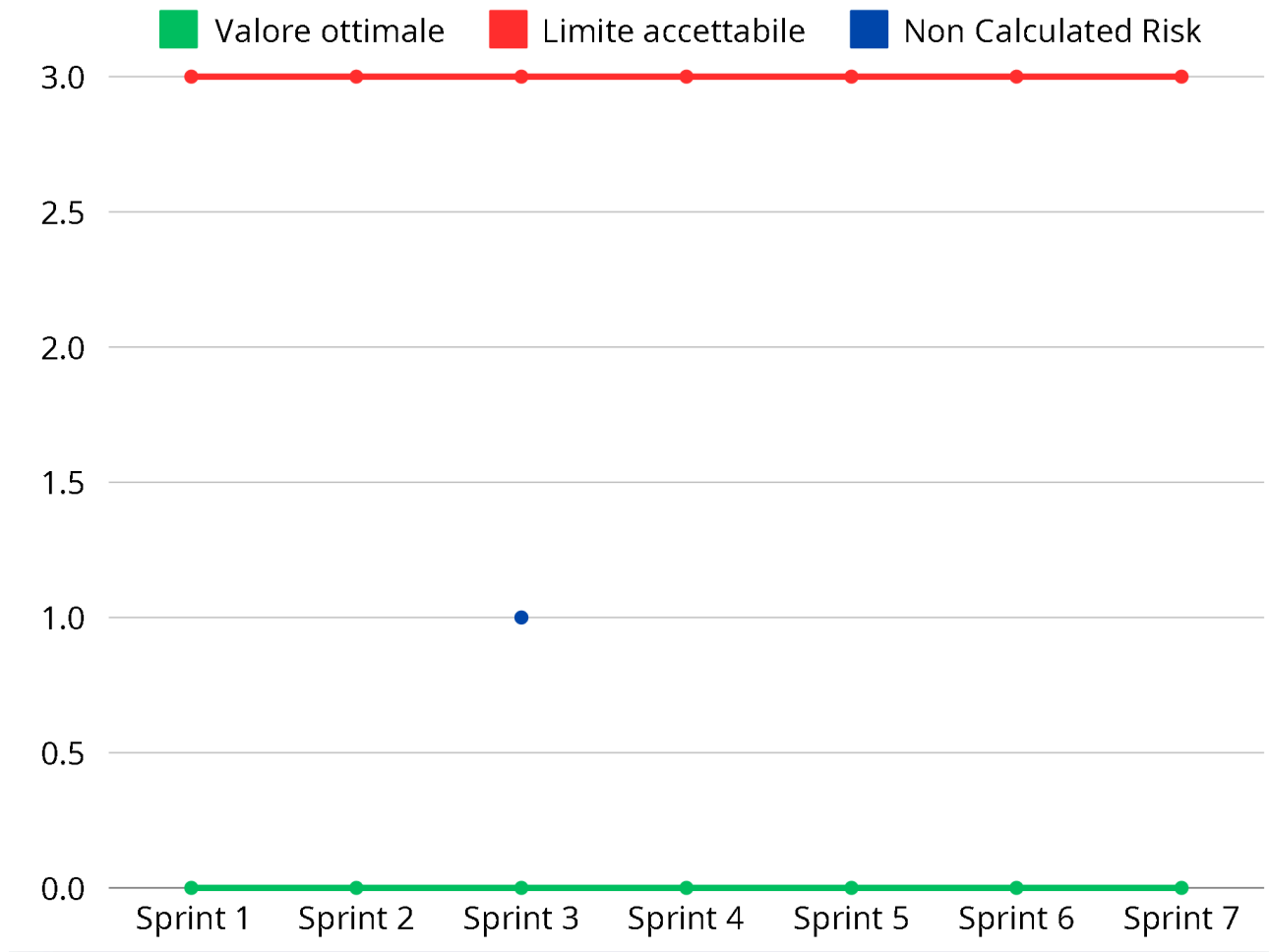


Figura 8: Stima dei valori di CV e SV durante i vari sprint.

Il grafico mostra come per la maggior parte degli sprint non si siano verificati rischi non calcolati, ma solo nello sprint 3 abbiamo avuto un problema di comunicazione interna al gruppo che ha richiesto un incontro con il professore Tullio Vardanega e che in seguito si è risolto. In ogni caso dal grafico si può notare che il team ha avuto una buona previsione dei rischi.

4.9. MPC-QMS(Quality Metric Satisfied)

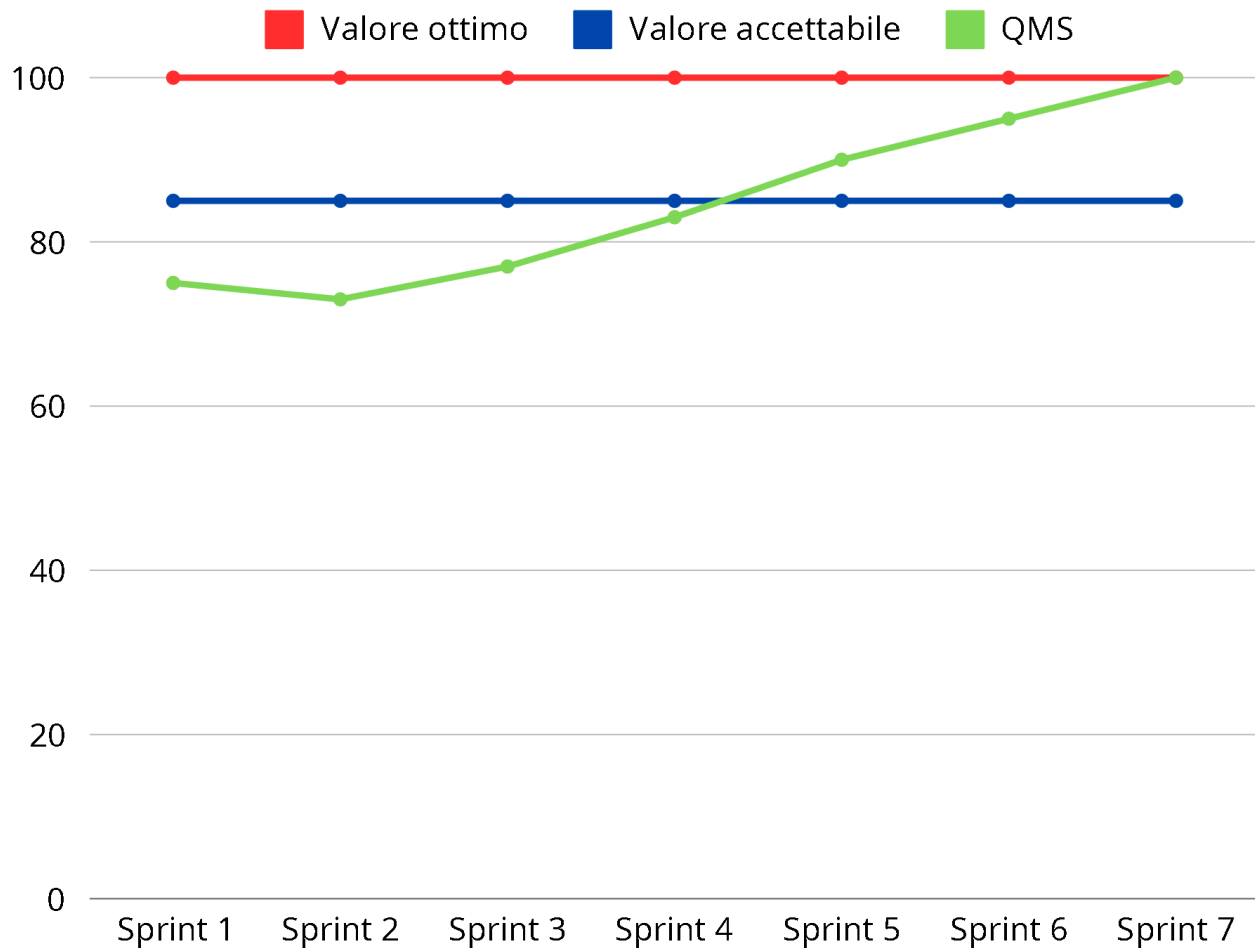


Figura 9: Stima della percentuale di metriche di qualità soddisfatte durante i vari sprint.

Dal grafico risulta che nei primi sprint una parte delle metriche di qualità non sono state soddisfatte o non hanno raggiunto un valore accettabile e questo è dovuto ad una iniziale inesperienza da parte dei membri del team, i quali, però, hanno capito questi errori apprendendo da essi e migliorando come gruppo. Questo fino a raggiungere valori accettabili e in seguito ottimi soprattutto negli ultimi sprint dimostrando un miglioramento nel [way of working*](#) e nei risultati ottenuti.

5. Processi di automiglioramento

5.1. Introduzione

In questa sezione vengono riportati tutti i problemi riscontrati dal gruppo durante lo svolgimento del progetto. Per ogni problema viene indicato il periodo in cui è stato riscontrato, la descrizione del problema, l'azione correttiva adottata.

5.2. Valutazione sull'organizzazione, pianificazione e incontri

Periodo	Problema	Azione correttiva
Tutto il progetto	È risultato molto difficile organizzare incontri dove fossero presenti tutti i membri del gruppo, per questioni di lavoro, orari, impegni personali, ed esami durante la sessione invernale	Il gruppo ha deciso di organizzare incontri basandosi sulle disponibilità dichiarate da ogni membro su un Foglio Google dedicato, nel caso alcuni membri non erano presenti, questi venivano informati sulle decisioni prese durante un incontro tramite verbali ed eventualmente dei messaggi su Telegram
Tutto il progetto	È risultato difficile cercare di rimanere entro i limiti di tempo e budget assegnati ad ogni ruolo	Per organizzarsi al meglio, il gruppo ha deciso di tenere un tabella su un Foglio Google che tiene conto delle ore individuali e totali di ogni ruolo
Tutto il progetto	È risultato difficile lavorare sugli stessi documenti nello stesso periodo di tempo senza che più persone scrivessero nella stessa sezione sovrascrivendo il lavoro di altri	Si è deciso di utilizzare diversi branch su git, ognuno dedicato ad ogni attività per ogni sprint, in questo modo è stato più facile trovare sezioni sovrascritte e verificare solo le parti cambiate del documento

Periodo	Problema	Azione correttiva
Inizio progetto	Nella fase iniziale si sono presentati dei problemi sulla formattazione del testo, struttura, stile e stesura dei documenti	Per organizzarsi meglio sulla stesura e stile dei documenti, si è fatto in modo che ogni documento prenda lo stile dallo stesso file, in modo da facilitarne anche la modifica
Inizio progetto	Alcuni lavori assegnati nella fase iniziale richiedevano troppo tempo per essere assegnati ad un solo membro	Il gruppo si è impegnato per fare in modo che le attività assegnate abbiano un livello di granularità più fine
Inizio progetto	Inizialmente venivano inserite nella branch main (principale) modifiche che presentavano errori, spesso a causa della mancanza di verifica	Il problema è stato risolto attraverso l'utilizzo di branch protection, che controlla se il verificatore ha approvato le modifiche, se la tabella delle versioni è correttamente compilata e se il sorgente del documento compila correttamente

Tabella 17: Valutazione sull'organizzazione

5.3. Valutazione sui ruoli

Periodo	Ruolo	Problema	Azione correttiva
Inizio progetto	Tutti	Il gruppo ha riscontrato difficoltà nel capire i compiti assegnati a ciascun ruolo, e come assegnarli	È stato deciso di assegnare i ruoli in maniera arbitraria in base alle attività da svolgere del singolo membro in caso di necessità

Periodo	Ruolo	Problema	Azione correttiva
Inizio progetto	Verificatore	Il ruolo del verificatore	Si è deciso di caricare ogni cambiamento alla repository attraverso una pull request, il verificatore ha il compito di approvarla
Inizio progetto	Responsabile, Amministratore	È risultata poco chiara la differenza tra i ruoli di responsabile e amministratore	I ruoli di responsabile e amministratore vengono dichiarati ad ogni riunione insieme agli altri ruoli, e vengono chiariti i singoli compiti da eseguire. Questi sono basati su quanto imparato dal corso di Ingegneria del Software

Tabella 18: Valutazione sui ruoli

5.4. Valutazione sugli strumenti

Periodo	Strumento	Problema	Azione correttiva
Inizio progetto	Git, GitHub	Non essendo molto familiare con gli strumenti di git, il gruppo ha avuto difficoltà all'inizio del progetto a gestire i branch, pull request e assegnare le issues	Durante un incontro sono state spiegate le diverse funzionalità di git per creare un branch nuovo e la procedura per il merge. Inoltre ogni issue veniva assegnato a una o più persone per

Periodo	Strumento	Problema	Azione correttiva
			garantire una migliore organizzazione
PoC	GPT-4o mini, Python, Svelte, MongoDB, FastAPI, Chroma	Essendo tutti strumenti nuovi per la maggior parte del gruppo, è risultato difficile capire come utilizzarli e come integrarli tra loro	Le persone già familiari con questi strumenti hanno spiegato il loro modo d'uso agli altri membri, sono state richieste comunque delle ore aggiuntive a ogni persona per imparare e capire le basi degli strumenti utilizzati
Fine primo periodo	Typst	Verso il revisionamento per RTB, la nuova versione di Typst 0.13 ha causato dei problemi nel compilare le tabelle. Queste venivano schiacciate e venivano visualizzate come una singola linea	Nella nuova versione è stato modificato un tag usato per creare le tabelle nei documenti, è bastato aggiungere un altro tag insieme a quello usato

Tabella 19: Valutazione sugli strumenti utilizzati