Piano di qualifica

v2.0

7Last

Versioni

Ver.	Data	Autore	Verificatore$_G$	Descrizione
1.0	2024-05-24	Matteo Tiozzo	Raul Seganfreddo	Approvazione finale documento
0.7	2024-05-20	Matteo Tiozzo	Valerio Occhinegro	Stesura iniziative di automiglioramento
0.6	2024-05-14	Leonardo Baldo	Antonio Benetazzo	Popolamento grafici Cruscotto$_G$ di valutazione della qualità
0.5	2024-05-17	Antonio Benetazzo	Davide Malgarise	Creazione grafici per Cruscotto$_G$
0.4	2024-04-22	Antonio Benetazzo	Davide Malgarise	Fine stesura metodologie di testing
0.3	2024-04-16	Valerio Occhinegro	Matteo Tiozzo	Inizio stesura metodologie di testing
0.2	2024-03-29	Valerio Occhinegro	Matteo Tiozzo	Stesura metriche di qualità
0.1	2024-03-28	Valerio Occhinegro	Matteo Tiozzo	Prima redazione

Indice

Elenco delle tabelle

Elenco delle figure

1 Introduzione

1.1 Obiettivo del documento

Il presente documento ha lo scopo di definire le strategie di verifica e validazione utilizzate per assicurare il corretto funzionamento e uno standard di qualità dello strumento sviluppato e delle attività che lo accompagnano. Sarà sottoposto a revisioni continue, così da prevedere situazioni precedentemente non occorse e da seguire l'evoluzione del progetto.

1.2 Glossario

Il glossario$_G$ è uno strumento utilizzato per risolvere eventuali dubbi riguardanti alcuni termini specifici utilizzati nella redazione del documento. Esso conterrà la definizione dei termini evidenziati e sarà consultabile al seguente link. I termini presenti in tale documento saranno evidenziati da una 'G' a pedice.

1.3 Riferimenti

1.3.1 Normativi

- **Regolamento del progetto**
  `https://www.math.unipd.it/~tullio/IS-1/2023/Dispense/PD2.pdf.`
- **Norme di progetto**$_G$ v2.0
  `https://7last.github.io/docs/rtb/documentazione-interna/norme-di-progetto`

1.3.2 Informativi

- **Standard ISO/IEC 25010:2023**
  `https://iso25000.com/index.php/en/iso-25000-standards/iso-25010`
- **Standard ISO/IEC 12207:1995**
  `https://www.math.unipd.it/~tullio/IS-1/2009/Approfondimenti/ISO_12207-1995.pdf`
- **Qualità di prodotto**
  `https://www.math.unipd.it/~tullio/IS-1/2023/Dispense/T7.pdf`
- **Qualità di processo**
  `https://www.math.unipd.it/~tullio/IS-1/2023/Dispense/T8.pdf`

- **Verifica e validazione**
  - Introduzione
    `https://www.math.unipd.it/~tullio/IS-1/2023/Dispense/T9.pdf`
  - Analisi statica
    `https://www.math.unipd.it/~tullio/IS-1/2023/Dispense/T10.pdf`
  - Analisi dinamica
    `https://www.math.unipd.it/~tullio/IS-1/2023/Dispense/T11.pdf`
- **Capitolato$_G$ d'appalto C6**: *SyncCity$_G$* – A smart city$_G$ monitoring platform
  `https://www.math.unipd.it/~tullio/IS-1/2023/Progetto/C6.pdf`
- **Verbali esterni**
  `https://7last.github.io/docs/category/verbali-esterni-1`
- **Verbali interni**
  `https://7last.github.io/docs/category/verbali-interni-1`
- **Analisi dei requisiti$_G$** v2.0
  `https://7last.io/docs/documentazione-esterna/analisi-dei-requisiti`
- **Glossario$_G$** v2.0
  `https://7last.github.io/docs/pb/documentazione-interna/glossario`

2 Metriche di qualità

La qualità di processo è un criterio fondamentale ed è alla base di ogni prodotto che rispecchi lo stato dell'arte. Per raggiungere tale obiettivo è necessario sfruttare delle pratiche rigorose che consentano lo svolgimento di ogni attività in maniera ottimale. Al fine di valutare nel miglior modo possibile la qualità del prodotto e l'efficacia dei processi, sono state definite delle metriche, meglio specificate nel documento *Norme di Progetto*$_G$ e qui di seguito riepilogate. Esse sono state suddivise utilizzando lo **standard ISO/IEC 12207:1995**, il quale separa i processi di ciclo di vita del software in processi di base e/o primari, processi di supporto e processi organizzativi.

2.1 Processi di base e/o primari

2.1.1 Fornitura

Codice	Nome	Ammissibile	Ottimo
1M-PV	Planned Value	$PV \geq 0$	$PV \leq BAC$
2M-EV	Earned Value	$EV \geq 0$	$EV \leq EAC$
3M-AC	Actual Cost	$AC \geq 0$	$AC \leq EAC$
4M-SV	Schedule Variance	$SV \geq -10\%$	$SV \geq 0\%$
5M-CV	Cost Variance	$CV \geq -10\%$	$CV \geq 0\%$
6M-CPI	Cost Performance Index	$CPI \geq 0.8$	$CPI \geq 1$
7M-SPI	Schedule Performance Index	$SPI \geq 0.8$	$SPI \geq 1$
8M-EAC	Estimate At Completion	$EAC \leq BAC + 5\%$	$EAC \leq BAC$
9M-ETC	Estimate To Complete	$ETC \geq 0$	$ETC \leq EAC$
10M-OTDR	On-Time Delivery Rate	$OTDR \geq 90\%$	$OTDR \geq 95\%$

Tabella 1: Metriche di qualità per il processo di fornitura

2.1.2 Sviluppo

2.1.2.1 Analisi dei requisiti

Codice	Nome	Ammissibile	Ottimo
11M-PRO	Percentuale Requisiti Obbligatori	$PRO \geq 100\%$	$PRO \geq 100\%$
12M-PRD	Percentuale Requisiti Desiderabili	$PRD \geq 35\%$	$PRD \geq 100\%$
13M-PRO	Percentuale Requisiti Opzionali	$PRO \geq 0\%$	$PRO \geq 100\%$

Tabella 2: Metriche di qualità per il processo di analisi dei requisiti$_G$

2.1.2.2 Progettazione

Codice	Nome	Ammissibile	Ottimo
14M-PG	Profondità delle Gerarchie	$PG \leq 7$	$PG \leq 5$

Tabella 3: Metriche di qualità per il processo di progettazione

2.1.2.3 Codifica

Codice	Nome	Ammissibile	Ottimo
15M-PPM	Parametri Per Metodo	$PPM \leq 7$	$PPM \leq 5$
16M-CPC	Campi Per Classe	$CPC \leq 8$	$CPC \leq 5$
17M-LCPM	Linee Di Commento Per Metodo	$LCPM \geq 50$	$LCPM \geq 20$
18M-CCM	Complessità Ciclomatica Media	$CCM \leq 6$	$CCM \leq 3$

Tabella 4: Metriche di qualità per il processo di codifica

2.2 Processi di supporto

2.2.1 Documentazione

Codice	Nome	Ammissibile	Ottimo
19M-IG	Indice Gulpease	$IG \geq 50$	$IG \geq 75$
20M-CO	Correttezza Ortografica	$CO = 0\ errori$	$CO = 0\ errori$

Tabella 5: Metriche di qualità per il processo di documentazione

2.2.2 Gestione della qualità

Codice	Nome	Ammissibile	Ottimo
21M-FU	Facilità di Utilizzo	$FU \geq 3\ errori$	$FU \geq 0\ errori$
22M-TA	Tempo di Apprendimento	$TA \leq 12\ min$	$TA \leq 7\ min$
23M-TR	Tempo di Risposta	$TR \leq 8\ sec$	$TR \leq 4\ sec$
24M-TE	Tempo di Elaborazione	$TE \leq 10\ sec$	$TE \leq 5\ sec$
25M-QMS	Metriche di Qualità Soddisfatte	$QMS \geq 90\%$	$QMS = 100\%$

Tabella 6: Metriche di qualità per il processo di gestione della qualità

2.2.3 Verifica

Codice	Nome	Ammissibile	Ottimo
26M-CC	Code Coverage	$CC \geq 80\%$	$CC \geq 100\%$
27M-BC	Branch Coverage	$BC \geq 80\%$	$BC \geq 100\%$
28M-SC	Statement Coverage	$SC \geq 80\%$	$SC \geq 100\%$
29M-FD	Failure Density	$FD \leq 15\%$	$FD = 0\%$
30M-PTCP	Passed Test Case Percentage	$PTCP \geq 90\%$	$PTCP \geq 100\%$

Tabella 7: Metriche di qualità per il processo di verifica

2.2.4 Risoluzione dei problemi

Codice	Nome	Ammissibile	Ottimo
31M-RMR	Risk Mitigation Rate	$RMR \geq 80\%$	$RMR \geq 100\%$
32M-NCR	Rischi Non Calcolati	$NCR \leq 3$	$NCR = 0$

Tabella 8: Metriche di qualità per il processo di risoluzione dei problemi

2.3 Processi organizzativi

2.3.1 Pianificazione

Codice	Nome	Ammissibile	Ottimo
33M-RSI	Requirements Stability Index	$RSI \geq 75\%$	$RSI = 100\%$

Tabella 9: Metriche di qualità per il processo di pianificazione

3 Metodologie di testing

In questa sezione verranno illustrate le metodologie di *testing* adottate per garantire il rispetto dei vincoli individuati nella sezione *Requisiti* del documento *Analisi dei Requisiti*$_G$. I test verranno suddivisi in cinque categorie:

- test di unità;
- test di integrazione;
- test di sistema;
- test di regressione;
- test di accettazione.

Verranno elencate le varie tipologie di test eseguite, indicando il codice del test, una breve descrizione di ciò che viene verificato e lo stato di avanzamento del test, espresso come segue:

- **S**: test superato;
- **NS**: test non superato;
- **S**: test non implementato.

3.1 Test di unità

I test di unità verificano il corretto funzionamento delle singole unità di codice, ovvero le più piccole parti di un programma, per assicurarsi che ognuna funzioni correttamente e che sia in grado di eseguire le operazioni richieste.

Codice	Descrizione	Stato
Python		
1T-U	Verificare che la classe `TemperatureRawData` venga creata correttamente.	S
2T-U	Verificare che il metodo `topic`$_G$`()` di `TemperatureRawData` restituisca `"temperature"`.	S

Codice	Descrizione	Stato
3T-U	Verificare che il metodo `subject()` di `TemperatureRawData` restituisca `"temperature-value"`.	S
4T-U	Verificare che la classe `TrafficRawData` venga creata correttamente.	S
5T-U	Verificare che il metodo $\texttt{topic}_{\texttt{G}}\texttt{()}$ di `TrafficRawData` restituisca `"traffic"`.	S
6T-U	Verificare che il metodo `subject()` di `TrafficRawData` restituisca `"traffic-value"`.	S
7T-U	Verificare che la classe `RecyclingPointRawData` venga creata correttamente.	S
8T-U	Verificare che il metodo $\texttt{topic}_{\texttt{G}}\texttt{()}$ di `RecyclingPointRawData` restituisca `"recycling_point"`.	S
9T-U	Verificare che il metodo `subject()` di `RecyclingPointRawData` restituisca `"recycling_point-value"`.	S
10T-U	Verificare che la classe `HumidityRawData` venga creata correttamente.	S
11T-U	Verificare che il metodo $\texttt{topic}_{\texttt{G}}\texttt{()}$ di `HumidityRawData` restituisca `"humidity"`.	S
12T-U	Verificare che il metodo `subject()` di `HumidityRawData` restituisca `"humidity-value"`.	S
13T-U	Verificare che la classe `AirQualityRawData` venga creata correttamente.	S
14T-U	Verificare che il metodo $\texttt{topic}_{\texttt{G}}\texttt{()}$ di `AirQualityRawData` restituisca `"air_quality"`.	S
15T-U	Verificare che il metodo `subject()` di `AirQualityRawData` restituisca `"air_quality-value"`.	S
16T-U	Verificare che la classe `RainRawData` venga creata correttamente.	S
17T-U	Verificare che il metodo $\texttt{topic}_{\texttt{G}}\texttt{()}$ di `RainRawData` restituisca `"rain"`.	S
18T-U	Verificare che il metodo `subject()` di `RainRawData` restituisca `"rain-value"`.	S

Codice	Descrizione	Stato
19T-U	Verificare che la classe `ChargingStationRawData` venga creata correttamente.	S
20T-U	Verificare che il metodo $\texttt{topic}_{\texttt{G}}\texttt{()}$ di `ChargingStationRawData` restituisca `"charging_station"`.	S
21T-U	Verificare che il metodo `subject()` di `ChargingStationRawData` restituisca `"charging_station-value"`.	S
22T-U	Verificare che la classe `ParkingLotRawData` venga creata correttamente.	S
23T-U	Verificare che il metodo $\texttt{topic}_{\texttt{G}}\texttt{()}$ di `ParkingLotRawData` restituisca `"parking_lot"`.	S
24T-U	Verificare che il metodo `subject()` di `ParkingLotRawData` restituisca `"parking_lot-value"`.	S
25T-U	Verificare che la classe `WaterLevelRawData` venga creata correttamente.	S
26T-U	Verificare che il metodo $\texttt{topic}_{\texttt{G}}\texttt{()}$ di `WaterLevelRawData` restituisca `"water_level"`.	S
27T-U	Verificare che il metodo `subject()` di `WaterLevelRawData` restituisca `"water_level-value"`.	S
28-U	Verificare che il metodo `from_str()` di `SensorType` effettui il parsing correttamente.	S
29-U	Verificare che la classe `EnvConfig` venga creata correttamente se tutte le variabili d'ambiente sono impostate.	S
30-U	Verificare che la classe `EnvConfig` venga creata correttamente se la variabile d'ambiente `MAX_BLOCK_MS` non è impostata.	S
31-U	Verificare che la creazione della classe `EnvConfig` fallisca con un'eccezione se le variabili d'ambiente non sono impostate.	S
32-U	Verificare che il metodo `bootstrap_server` della classe `EnvConfig` ritorni correttamente il valore dell'host concatenato alla porta con ':'..	S

Codice	Descrizione	Stato
33-U	Verificare che la classe `SensorConfig` sia creata correttamente.	S
34-U	Verificare che la creazione della classe `SensorConfig` fallisca con un'eccezione se il tipo di sensore$_G$ fornito non esiste.	S
35-U	Verificare che la creazione della classe `SensorConfig` fallisca con un'eccezione se il tipo di sensore$_G$ non è fornito.	S
36-U	Verificare che la creazione della classe `SensorConfig` fallisca con un'eccezione se il campo `generation_delay` non rispetta lo standard ISO8601.	S
37-U	Verificare che la creazione della classe `SensorConfig` fallisca con un'eccezione se il campo `points_spacing` non rispetta lo standard ISO8601.	S
38-U	Verificare che il metodo `data()` della classe `TemperatureSimulator` generi correttamente i dati casuali.	S
39-U	Verificare che il metodo `data()` della classe `TrafficSimulator` generi correttamente i dati casuali.	S
40-U	Verificare che il metodo `data()` della classe `RecyclingPointSimulator` generi correttamente i dati casuali.	S
41-U	Verificare che il metodo `data()` della classe `HumiditySimulator` generi correttamente i dati casuali.	S
42-U	Verificare che il metodo `data()` della classe `AirQualitySimulator` generi correttamente i dati casuali.	S
43-U	Verificare che il metodo `data()` della classe `RainSimulator` generi correttamente i dati casuali.	S
44-U	Verificare che il metodo `data()` della classe `ChargingStationSimulator` generi correttamente i dati casuali.	S

Codice	Descrizione	Stato
45-U	Verificare che il metodo `data()` della classe `ParkingLotSimulator` generi correttamente i dati casuali.	S
46-U	Verificare che il metodo `data()` della classe `WaterLevelSimulator` generi correttamente i dati casuali.	S
47-U	Verificare che il metodo `serialize(data)` della classe `AvroSerializationStrategy` serializzi correttamente i dati in formato Confluent Avro.	S
48-U	Verificare che il metodo `serialize(data)` della classe `JsonSerializationStrategy` serializzi correttamente i dati in formato JSON.	S
49-U	Verificare che il metodo `build(name, config)` della classe `SimulatorFactory` restituisca un'istanza del simulatore corretto.	S
50-U	Verificare che il metodo `produce(raw_data)` della classe `Producer` invii correttamente i dati alla destinazione.	S
Apache Flink		
51-U	Verificare che la classe `AverageWindowFunction` calcoli correttamente la media dei dati in input.	S
52-U	Verificare che la classe `ChargingEfficiencyJoinFunction` calcoli correttamente l'efficienza di ricarica nel caso in cui il parcheggio non sia mai stato occupato.	S
53-U	Verificare che la classe `ChargingEfficiencyJoinFunction` calcoli correttamente l'efficienza di ricarica nel caso in cui si ricevano dei dati non validi per cui la colonnina è stata utilizza ma il parcheggio non è mai stato occupato.	S

Codice	Descrizione	Stato
54-U	Verificare che la classe `ChargingEfficiencyJoinFunction` calcoli correttamente l'efficienza di ricarica nel caso in cui l'*efficiency rate* sia maggiore dell'*utilization rate*.	S
55-U	Verificare che la classe `ChargingEfficiencyJoinFunction` calcoli correttamente l'efficienza di ricarica nel caso in cui l'*utilization rate* sia maggiore dell'*efficiency rate*.	S
56-U	Verificare che la classe `ChargingEfficiencyJoinFunction` calcoli correttamente l'efficienza di ricarica nel caso in cui la colonnina non sia mai stata utilizzata.	S
57-U	Verificare che la classe `HeatIndexJoinFunction` calcoli correttamente l'indice di calore.	S
58-U	Verificare che la classe `TimeDifferenceAggregateFunction` calcoli il tempo totale in cui una colonnina di ricarica è stata utilizzata.	S
59-U	Verificare che la classe `TimeDifferenceAggregateFunction` calcoli il tempo totale in cui un parcheggio è stata utilizzato.	S

Tabella 10: Test di unità

3.2 Test di integrazione

I test di integrazione verificano il corretto funzionamento delle interfacce tra le varie unità di codice, assicurandosi che esse interagiscano correttamente tra di loro e che siano in grado di comunicare e scambiarsi i dati necessari.

Codice	Descrizione	Stato
	Python	
1T-I	Verificare che i dati generati dal sensore$_G$ di temperatura siano pubblicati correttamente nel rispettivo topic$_G$ Redpanda.	S
2T-I	Verificare che i dati generati dal sensore$_G$ di traffico siano pubblicati correttamente nel rispettivo topic$_G$ Redpanda.	S
3T-I	Verificare che i dati generati dal sensore$_G$ di isola ecologica siano pubblicati correttamente nel rispettivo topic$_G$ Redpanda.	S
4T-I	Verificare che i dati generati dal sensore$_G$ di umidità siano pubblicati correttamente nel rispettivo topic$_G$ Redpanda.	S
5T-I	Verificare che i dati generati dal sensore$_G$ di qualità dell'aria siano pubblicati correttamente nel rispettivo topic$_G$ Redpanda.	S
6T-I	Verificare che i dati generati dal sensore$_G$ di precipitazioni siano pubblicati correttamente nel rispettivo topic$_G$ Redpanda.	S
7T-I	Verificare che i dati generati dalle colonnine di ricarica siano pubblicati correttamente nel rispettivo topic$_G$ Redpanda.	S
8T-I	Verificare che i dati generati dai sensori di occupazione di parcheggi siano pubblicati correttamente nel rispettivo topic$_G$ Redpanda.	S
9T-I	Verificare che i dati generati dai sensori di livello dell'acqua siano pubblicati correttamente nel rispettivo topic$_G$ Redpanda.	S

Codice	Descrizione	Stato
10T-I	Verificare che i dati generati dal sensore$_G$ di temperatura siano memorizzati correttamente nel database.	S
11T-I	Verificare che i dati generati dal sensore$_G$ di temperatura aggregati per 5 minuti siano memorizzati correttamente nel database.	S
12T-I	Verificare che i dati generati dal sensore$_G$ di temperatura aggregati per settimana siano memorizzati correttamente nel database.	S
13T-I	Verificare che i dati generati dal sensore$_G$ di temperatura aggregati per giorno siano memorizzati correttamente nel database.	S
14T-I	Verificare che i dati generati dal sensore$_G$ di traffico siano memorizzati correttamente nel database.	S
15T-I	Verificare che i dati generati dal sensore$_G$ di traffico aggregati per 5 minuti siano memorizzati correttamente nel database.	S
16T-I	Verificare che i dati generati dal sensore$_G$ di traffico aggregati per ora siano memorizzati correttamente nel database.	S
17T-I	Verificare che i dati generati dal sensore$_G$ di isola ecologica siano memorizzati correttamente nel database.	S
18T-I	Verificare che i dati generati dal sensore$_G$ di isola ecologica aggregati per 5 minuti siano memorizzati correttamente nel database.	S
19T-I	Verificare che i dati generati dal sensore$_G$ di umidità siano memorizzati correttamente nel database.	S
20T-I	Verificare che i dati generati dal sensore$_G$ di qualità dell'aria siano memorizzati correttamente nel database.	S
21T-I	Verificare che i dati generati dal sensore$_G$ di precipitazioni siano memorizzati correttamente nel database.	S

Codice	Descrizione	Stato
22T-I	Verificare che i dati generati dalle colonnine di ricarica siano memorizzati correttamente nel database.	S
23T-I	Verificare che i dati generati dai sensori di occupazione di parcheggi siano memorizzati correttamente nel database.	S
24T-I	Verificare che i dati generati dai sensori di livello dell'acqua siano memorizzati correttamente nel database.	S
25T-I	Verificare che i dati salvati su Clickhouse$_G$ siano correttamente accessibili da Grafana$_G$.	S
	Apache Flink	
26T-I	Verificare che il *job* `ChargingEfficiencyJob` calcoli correttamente l'efficienza di ricarica per un singolo sensore.	S
27T-I	Verificare che il *job* `ChargingEfficiencyJob` calcoli correttamente l'efficienza di ricarica per più sensori.	S
28T-I	Verificare che il *job* `ChargingEfficiencyJob` calcoli correttamente l'efficienza nel caso di efficienza e utilizzo massimi.	S
29T-I	Verificare che il *job* `ChargingEfficiencyJob` calcoli correttamente l'efficienza nel caso di utilizzo massimo e efficienza minima.	S
30T-I	Verificare che il *job* `HeatIndexJob` calcoli correttamente l'indice di calore per un singolo sensore.	S
31T-I	Verificare che il *job* `HeatIndexJob` calcoli correttamente l'indice di calore per più sensori.	S

Tabella 11: Test di integrazione

3.3 Test di sistema

I test di sistema sono finalizzati alla verifica del soddisfacimento dei requisiti richiesti ed evidenziati nel documento *Analisi dei Requisiti*$_G$. Questi test vengono effettuati sul sistema nel suo complesso, per verificare che il software funzioni correttamente e che sia in grado di eseguire le operazioni richieste.

Codice	Descrizione	Stato
1T-S	Verificare che l'accesso al sistema non richieda alcuna procedura di login e che sia direttamente accessibile dall'utente.	S
2T-S	Verificare che il prodotto non abbia alcuna sezione o funzionalità di amministrazione o gestione riservata.	S
3T-S	Verificare che i sensori integrati producano una misurazione coerente con il tipo di sensore$_G$ simulato.	S
4T-S	Verificare che ogni misurazione inviata dal simulatore contenga l'identificativo del sensore$_G$, le misurazioni d'interesse e il timestamp.	S
5T-S	Verificare che il sistema sia in grado di ricevere e memorizzare correttamente le misurazioni inviate dai sensori.	S
6T-S	Verificare che il sistema sia in grado di simulare almeno un sensore$_G$ per rilevare la temperatura.	S
7T-S	Verificare che il sistema sia in grado di simulare almeno un sensore$_G$ per rilevare il traffico.	S
8T-S	Verificare che il sistema sia in grado di simulare almeno un sensore$_G$ per rilevare il riempimento delle isole ecologiche.	S
9T-S	Verificare che il sistema sia in grado di simulare almeno un sensore$_G$ per rilevare l'umidità.	S
10T-S	Verificare che il sistema sia in grado di simulare almeno un sensore$_G$ per rilevare la qualità dell'aria.	S

Codice	Descrizione	Stato
11T-S	Verificare che il sistema sia in grado di simulare almeno un sensore$_G$ per rilevare le precipitazioni.	S
12T-S	Verificare che il sistema sia in grado di simulare almeno un sensore$_G$ per rilevare le colonnine di ricarica.	S
13T-S	Verificare che il sistema sia in grado di simulare almeno un sensore$_G$ per rilevare l'occupazione dei parcheggi.	S
14T-S	Verificare che il sistema sia in grado di simulare almeno un sensore$_G$ per rilevare il livello dell'acqua.	S
14T-S	Verificare che ogni dato generato dai simulatori dei sensori sia strettamente correlato al dato successivo, garantendo una transizione realistica tra le misurazioni.	S
15T-S	Verificare la facilità di comprensione e l'intuitività dell'interfaccia grafica, garantendo un'esperienza utente piacevole e soddisfacente.	S
16T-S	Verificare che le dashboard$_G$ si aggiornino quasi istantaneamente per riflettere i dati provenienti dai sensori entro un massimo di 15 secondi.	S
17T-S	Verificare che la dashboard$_G$ del traffico contenga almeno un *panel*$_G$ con un grafico time-series.	S
18T-S	Verificare che la dashboard$_G$ della temperatura contenga almeno un *panel*$_G$ con un grafico time-series.	S
19T-S	Verificare che la dashboard$_G$ delle isole ecologiche contenga almeno un *panel*$_G$ con un grafico time-series.	S
20T-S	Verificare che la dashboard$_G$ dell'umidità contenga almeno un *panel*$_G$ con un grafico time-series.	S
21T-S	Verificare che la dashboard$_G$ della qualità dell'aria contenga almeno un *panel*$_G$ con un grafico time-series.	S

Codice	Descrizione	Stato
22T-S	Verificare che la dashboard$_G$ delle precipitazioni contenga almeno un *panel*$_G$ con un grafico time-series.	S
23T-S	Verificare che la dashboard$_G$ dei parcheggi contenga almeno un *panel*$_G$ con un grafico time-series.	S
24T-S	Verificare che la dashboard$_G$ delle colonnine di ricarica contenga almeno un *panel*$_G$ con un grafico time-series.	S
25T-S	Verificare che la dashboard$_G$ del livello di acqua contenga almeno un *panel*$_G$ con un grafico time-series.	S
26T-S	Verificare che la dashboard$_G$ delle isole ecologiche contenga almeno un *panel*$_G$ con un grafico time-series.	S
27T-S	Verificare che i sensori presenti sulla mappa siano distinguibili in modo chiaro ed inequivocabile, permettendo il riconoscimento della loro tipologia.	S
28T-S	Verificare che in ciascuna dashboard$_G$ l'utente possa filtrare la visualizzazione delle misurazioni di uno specifico sensore$_G$.	S
29T-S	Verificare che nella dashboard$_G$ dei dati grezzi l'utente possa visualizzare la lista delle misurazioni in un formato tabellare, divise per tipo di sensore$_G$.	S
30T-S	Verificare che l'utente riceva notifiche quando i sensori superano determinate soglie di sicurezza.	S
31T-S	Verificare che l'utente possa visualizzare correttamente le coordinate dei sensori, con un numero congruo di cifre decimali.	S
32T-S	Verificare che l'utente possa visualizzare correttamente l'unità di misura associata a ciascuna misurazione.	S

Codice	Descrizione	Stato
33T-S	Verificare che nella dashboard$_G$ dei dati grezzi l'utente possa visualizzare una tabella contente l'indentificativo del sensore$_G$, la sua tipologia e la data dell'ultimo messaggio da esso inviato.	S

Tabella 12: Test di sistema

3.4 Test di accettazione

I test di accettazione vengono effettuati per verificare che il software soddisfi i requisiti richiesti e consentono di ultimare il processo di validazione del prodotto finale. Essi verranno eseguiti sia dal gruppo di sviluppo *7Last* che dall'azienda proponente$_G$ *SyncLab S.r.l.*.

Codice	Descrizione	Stato
1T-A	Verificare che tutti i widget$_G$ relativi alle diverse tipologie di sensori siano visibili sulla dashboard$_G$.	S
2T-A	Verificare che la mappa dei sensori si carichi correttamente e permetta interazioni fluide.	S
3T-A	Verifica della gestione corretta degli errori nel caso in cui i dati dei sensori non siano disponibili.	S
4T-A	Verifica della corretta visualizzazione delle misurazioni effettuate nel tempo dai sensori.	S
6T-A	Verificare che sia possibile visualizzare correttamente la dashboard$_G$ dei sensori di temperatura.	S
7T-A	Verificare che sia possibile visualizzare correttamente la dashboard$_G$ dei sensori di traffico.	S
8T-A	Verificare che sia possibile visualizzare correttamente la dashboard$_G$ dei sensori di isola ecologica.	S
9T-A	Verificare che sia possibile visualizzare correttamente la dashboard$_G$ dei sensori di umidità.	S

Codice	Descrizione	Stato
10T-A	Verificare che sia possibile visualizzare correttamente la dashboard$_G$ dei sensori di qualità dell'aria.	S
11T-A	Verificare che sia possibile visualizzare correttamente la dashboard$_G$ dei sensori di precipitazioni.	S
12T-A	Verificare che sia possibile visualizzare correttamente la dashboard$_G$ dei sensori di colonnine di ricarica.	S
13T-A	Verificare che sia possibile visualizzare correttamente la dashboard$_G$ dei sensori di occupazione di parcheggi.	S
14T-A	Verificare che sia possibile visualizzare correttamente la dashboard$_G$ dei sensori di livello dell'acqua.	S
15T-A	Verificare che sia possibile visualizzare correttamente la dashboard$_G$ dei dati grezzi	S
16T-A	Verificare si possa filtrare correttamente la visualizzazione delle misurazioni in base al sensore$_G$ che le ha prodotte.	S
17T-A	Verificare che si possa rimuovere correttamente i filtri attivi per visualizzazione delle misurazioni dei sensori.	S
18T-A	Verificare che si riceva correttamente una notifica in caso di superamento delle soglie impostate per le misurazioni.	S

Tabella 13: Test di accettazione

4 Cruscotto di valutazione della qualità

4.1 Qualità del processo di fornitura

4.1.1 1M-PV - Planned value e 2M-EV - Earned value

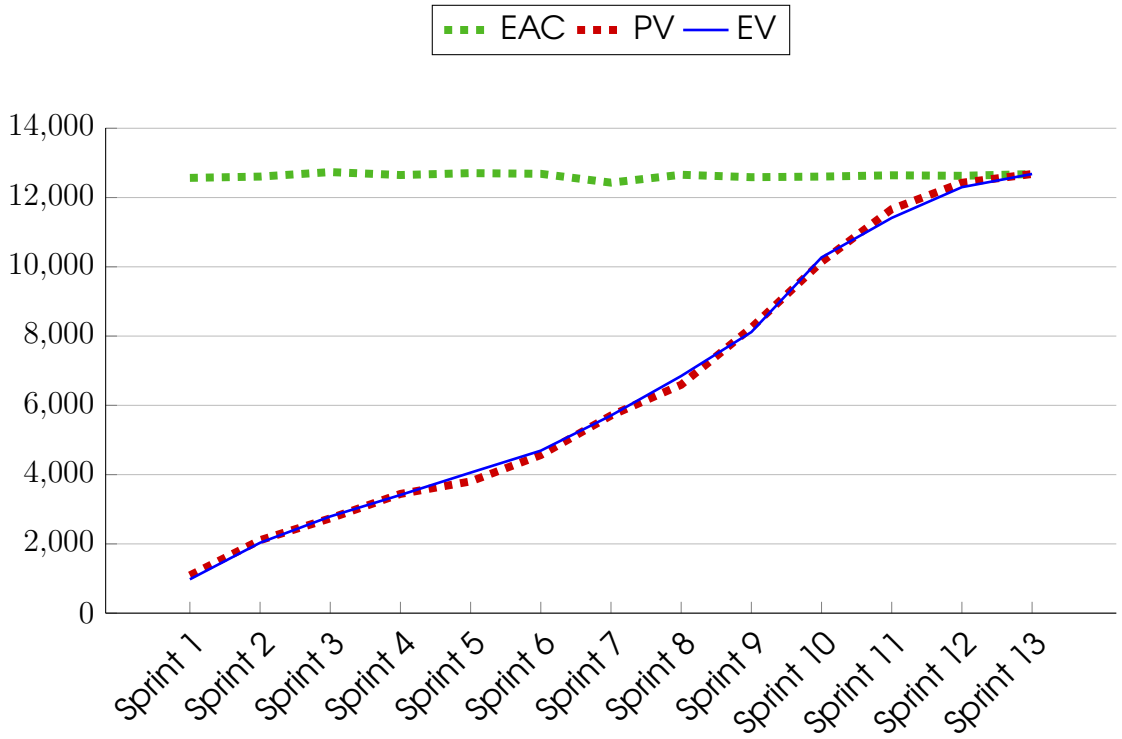

Figura 1: Proiezione del PV e dell'EV

RTB

Visionando il grafico si può notare che i valori di EV e PV quasi si sovrappongono, questo indica la buona riuscita della pianificazione delle attività da parte del gruppo *7Last*.

PB

Dal grafico completo si evince che il gruppo ha svolto le attività in modo efficiente, con un'evoluzione costante e in linea con la pianificazione. Questo è supportato dal fatto che i valori di EV e PV sono molto vicini tra loro, indicando che il gruppo ha pianificato correttamente le attività, confermando l'andamento positivo del progetto.

4.1.2 3M-AC - Actual cost e 9M-ETC - Estimate to complete

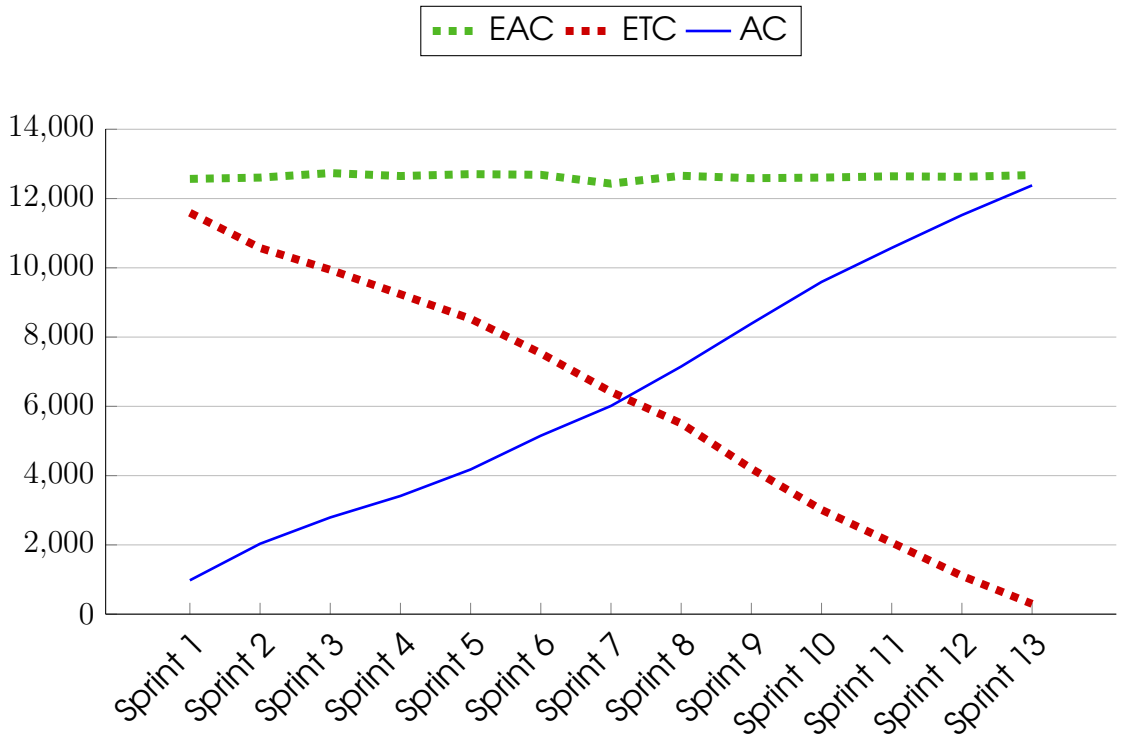

Figura 2: Proiezione dell'AC e dell'ETC

RTB

Il grafico evidenzia chiaramente un aumento progressivo dei costi (AC). Parallelamente, si osserva una diminuzione della stima dei costi a finire (ETC), che sta calando in modo proporzionale all'incremento dei costi.

PB

In questo secondo periodo viene confermato l'andamento evidenziato in quello precedente, con i costi reali (AC) che aumentano in modo proporzionale alla diminuzione della stima dei costi a finire (ETC). Questo indica che il gruppo ha svolto le attività in modo efficiente, con un allineamento tra i costi reali e quelli preventivati.

4.1.3 4M-SV - Schedule variance e 5M-CV - Cost variance

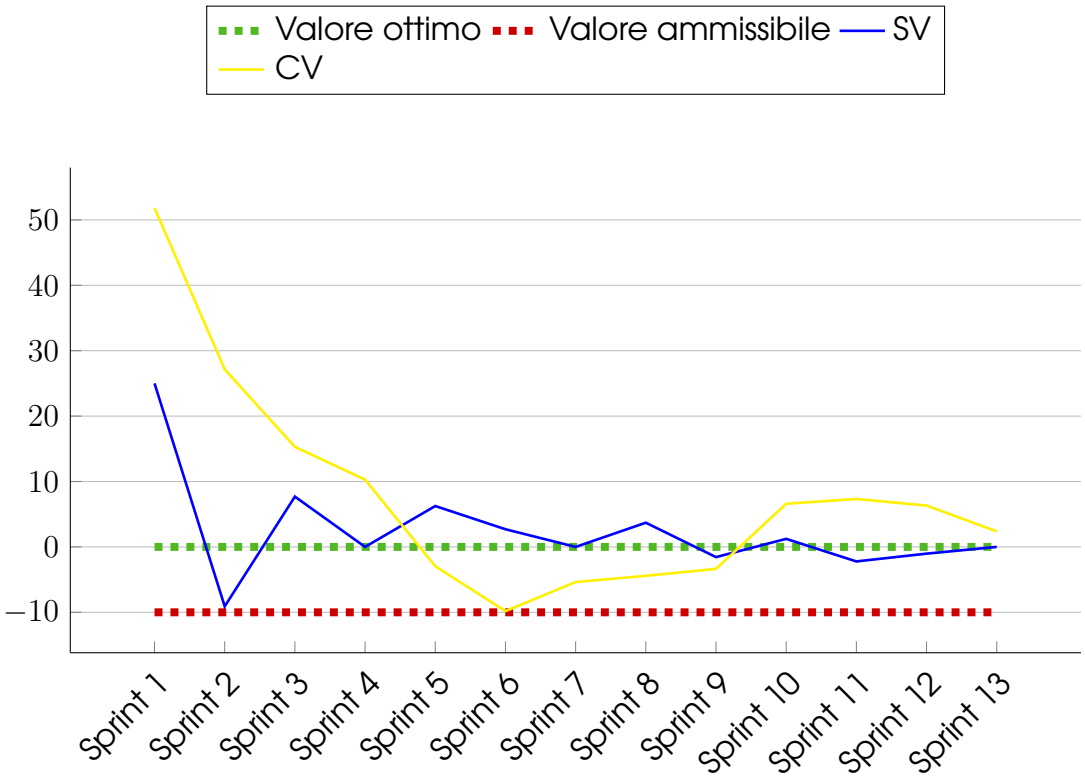

Figura 3: Andamento percentuale di SV e CV

RTB

Dal grafico si nota come sia SV che CV siano inizialmente elevati, per poi decrescere durante la prosecuzione del progetto, in particolare si nota un andamento altalenante del SV. L'andamento inizialmente alto del Schedule Variance (SV) e del Cost Variance (CV) indica una possibile sovrastima iniziale dei tempi e dei costi, dovuta all'inesperienza del team. La variabilità del SV suggerisce che le stime di tempistiche iniziali erano eccessivamente conservative, con aggiustamenti successivi man mano che il team acquisiva esperienza. La decrescita nel tempo di entrambe le metriche mostra che il gruppo sta diventando più preciso nelle sue previsioni, con un allineamento progressivo dei costi e delle tempistiche reali rispetto a quelle pianificate.

PB

L'andamento del grafico nella seconda parte del progetto conferma il trend positivo iniziato nel periodo precedente, con SV e CV che si avvicinano ai valori ottimali.

4.1.4 8M-EAC - Estimated at completion

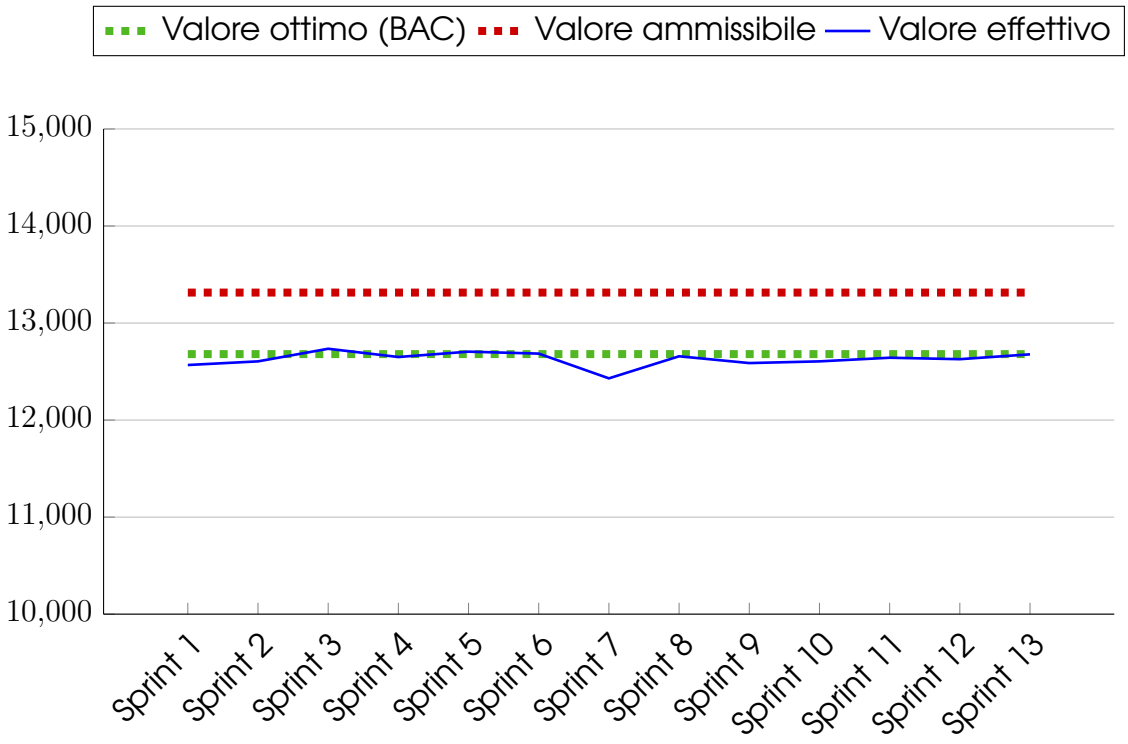

Figura 4: Proiezione dell'EAC

RTB

Osservando il grafico si può notare come l'EAC sia quasi sovrapposto al BAC durante i periodi di progetto analizzati fino ad ora. Questa situazione riflette come *7Last* abbia attuato una gestione efficace sia dei costi che delle tempistiche durante i periodi analizzati fino ad ora.

PB

Anche in questo grafico abbiamo una conferma dell'andamento rilevato nella prima parte del progetto, con l'EAC che si mantiene costantemente vicino al BAC. Questo indica che il gruppo ha svolto le attività in modo efficiente, con un allineamento tra i costi reali e quelli preventivati, senza variazioni significative.

4.2 Qualità del processo di analisi dei requisiti

4.2.1 11M-PRO - Percentuale requisiti obbligatori

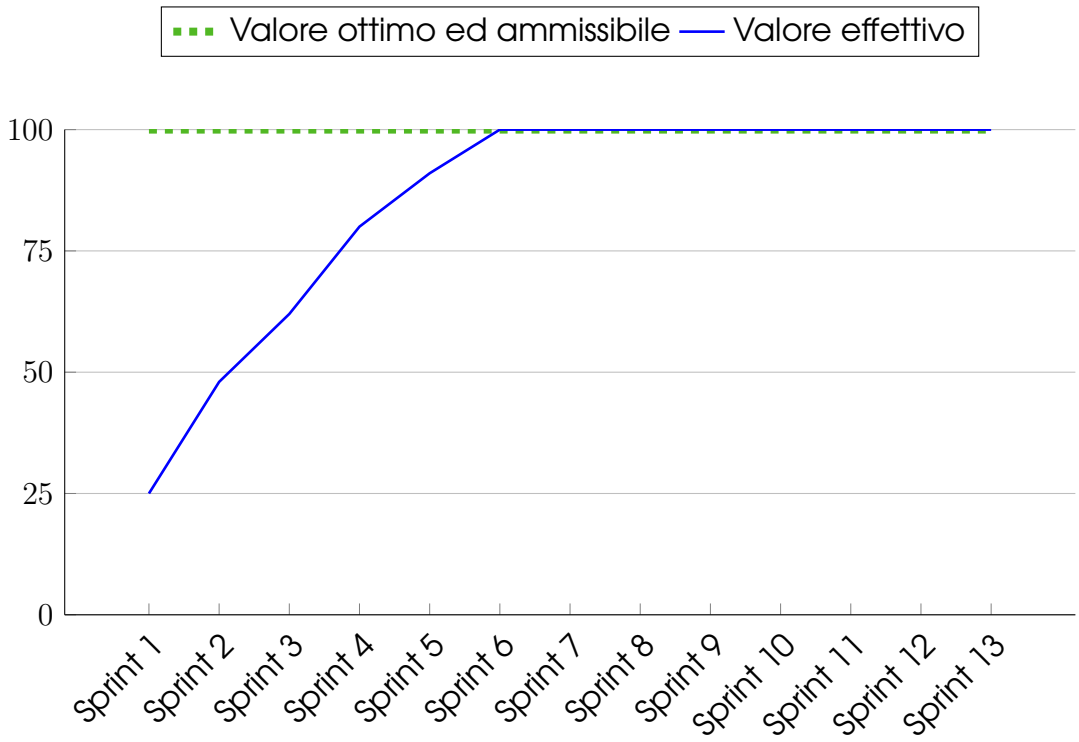

Figura 5: Percentuale di copertura dei requisiti obbligatori

PB

Dal grafico si evince come a partire dai primi sprint il gruppo *7Last* abbia lavorato in modo costante per soddisfare i requisiti obbligatori. Questo è confermato dal fatto che la percentuale di requisiti obbligatori soddisfatti è sempre cresciuta, fino al raggiungimento del 100% già a partire dal sesto sprint, confermando che il gruppo ha lavorato in modo efficace e con un'attenzione costante ai requisiti obbligatori.

4.2.2 12M-PRD - Percentuale requisiti desiderabili

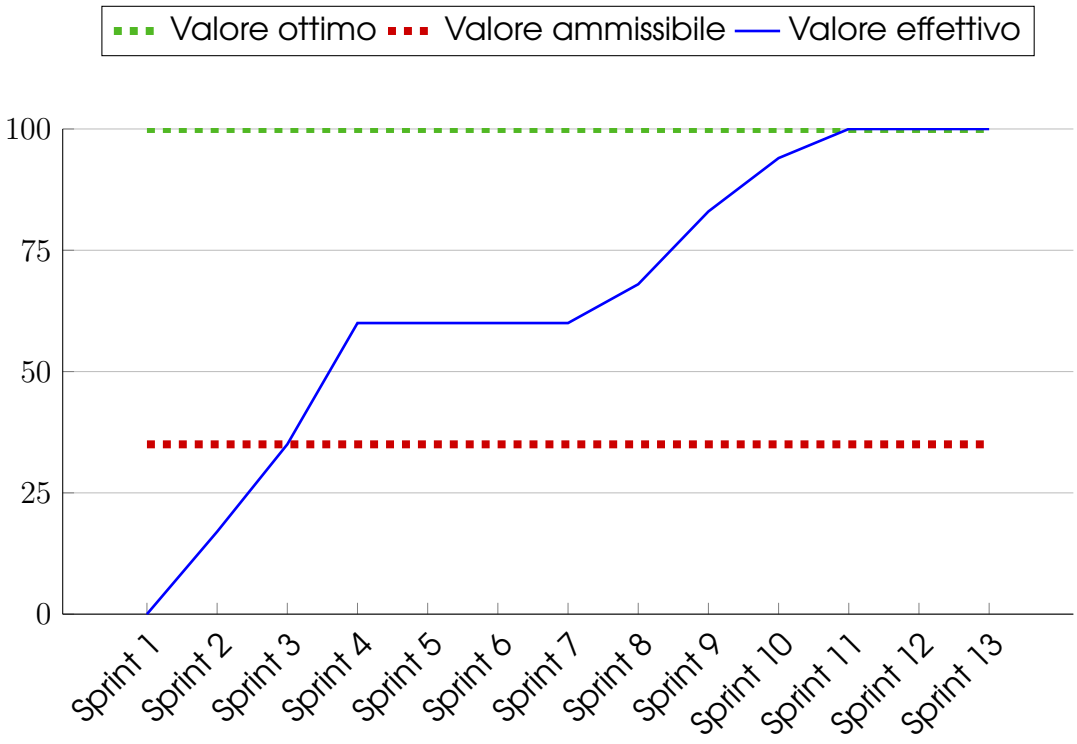

Figura 6: Percentuale di copertura dei requisiti desiderabili

PB

Il grafico mostra come il gruppo *7Last* abbia lavorato per soddisfare i requisiti desiderabili, raggiungendo l'ottimo risultato del 100% di copertura. Questo conferma che il gruppo ha lavorato in modo efficace e con un'attenzione costante ai requisiti desiderabili.

4.2.3 13M-PRO - Percentuale requisiti opzionali

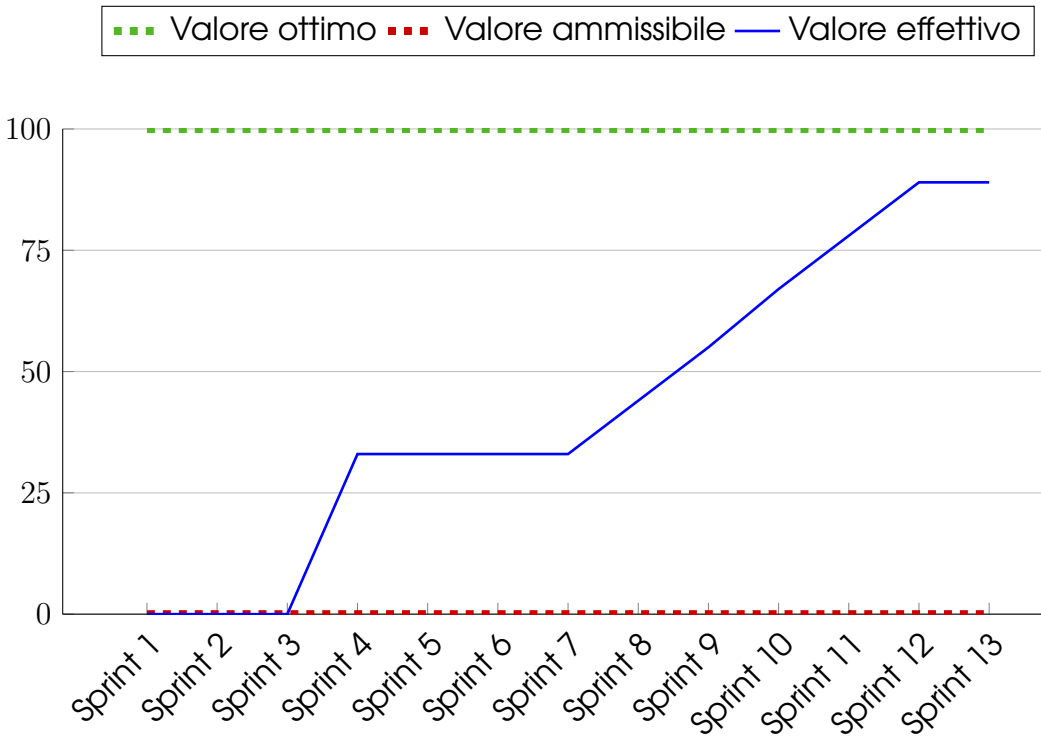

Figura 7: Percentuale di copertura dei requisiti opzionali

PB

Il grafico mostra come il gruppo *7Last* abbia lavorato per soddisfare i requisiti opzionali, con una crescita quasi costante della percentuale di requisiti soddisfatti. Questo conferma che il gruppo ha lavorato in modo efficace.

4.3 Qualità del processo di documentazione

4.3.1 19M-IG - Indice di Gulpease

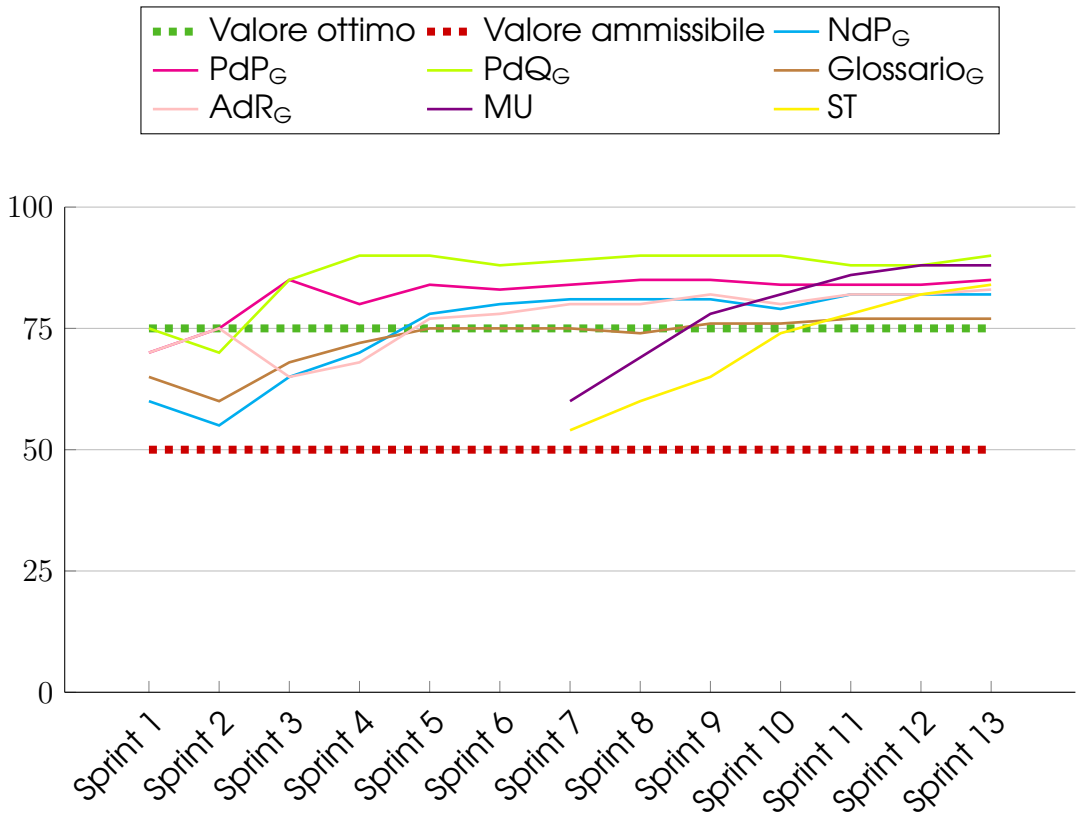

Figura 8: Andamento indice di Gulpease per ciascun documento

RTB

Visionando il grafico si può notare una tendenza generale di crescita, eccetto per alcuni documenti. L'indice relativamente basso rispetto agli altri documenti rappresenta il glossario$_G$, il quale contiene descrizioni di natura tecnica che possono influire negativamente sull'indice di Gulpease.

PB

Il grafico mostra come la qualità della documentazione prodotta verso la fine del progetto sia sempre sopra il valore ottimo prestabilito, con un andamento generale di crescita.

4.3.2 20M-CO - Correttezza ortografica

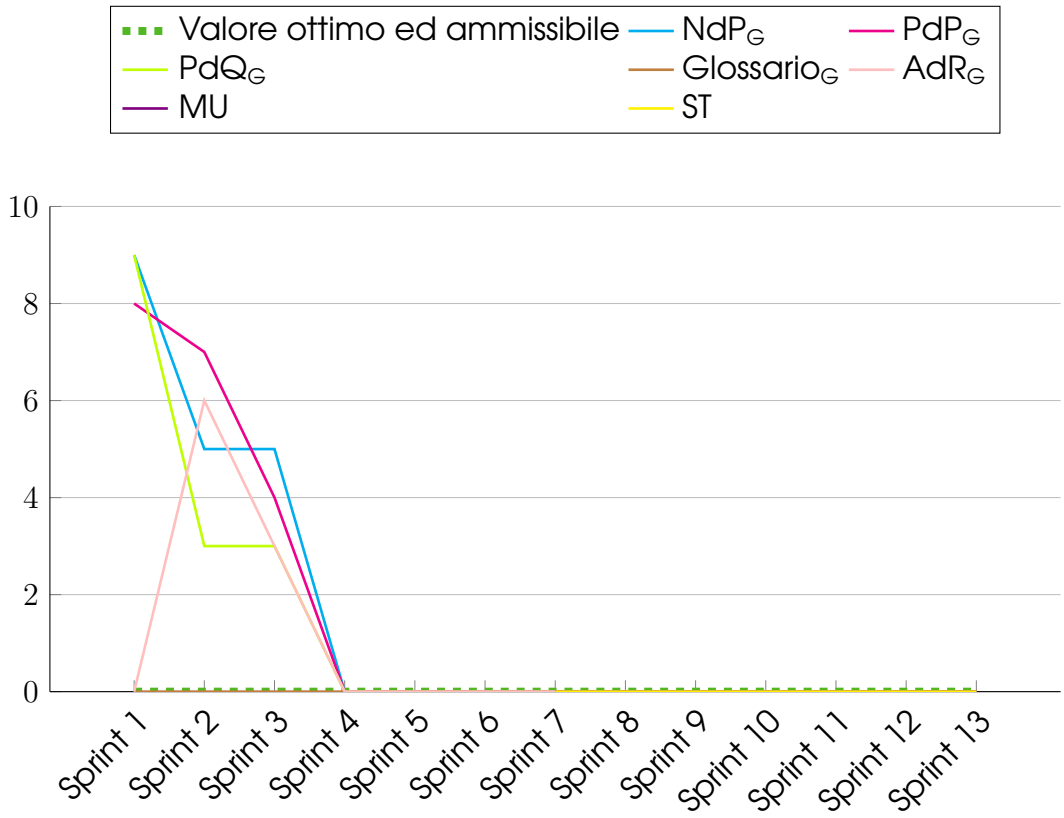

Figura 9: Errori ortografici per ciascun documento

RTB

Si noti come inizialmente il numero di errori di ortografia rilevati nei documenti sia elevato, per poi diminuire progressivamente. Questo indica che il gruppo *7Last* ha migliorato la qualità della documentazione prodotta, riducendo gli errori di ortografia.

PB

Come si può notare dal grafico a partire dal quarto sprint la documentazione prodotta non presenta più errori di ortografia. Questo è dovuto all'utilizzo di strumenti di controllo ortografico automatici in fase di *merge* dei branch nel repository, gestiti mediante l'utilizzo delle *GitHub Actions*. Questo ha permesso di impedire in modo sistematico l'inserimento di errori di ortografia nei documenti prodotti.

4.4 Qualità del processo di gestione della qualità

4.4.1 25M-QMS - Metriche di qualità soddisfatte

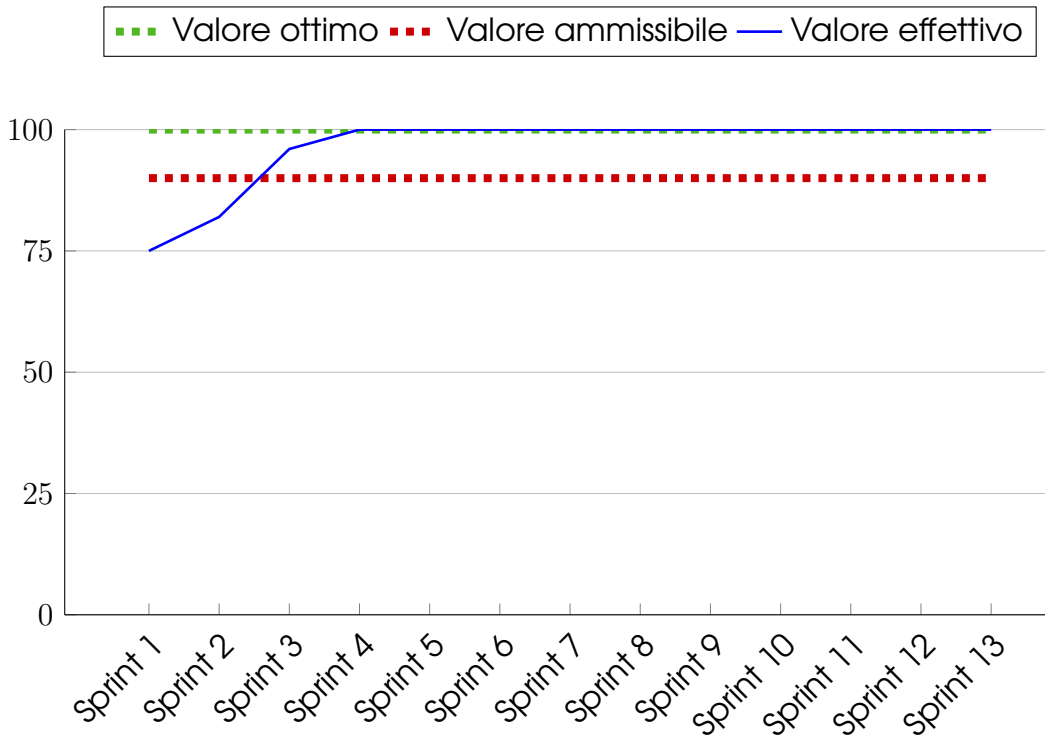

Figura 10: Percentuale di metriche di qualità soddisfatte

RTB

Osservando il grafico si può notare come inizialmente il valore delle metriche soddisfatte sia inferiore al valore ammissibile, questo è dovuto principalmente all'inesperienza del team. Successivamente l'andamento cresce progressivamente fino ad arrivare al 100% nell'ultimo sprint. Questo indica un miglioramento progressivo del *Way of Working* del gruppo.

PB

Come si evince dal grafico, il cruscotto di valutazione della qualità ha permesso al gruppo di monitorare costantemente il soddisfacimento delle metriche di qualità, ottenendo così un miglioramento progressivo fino al raggiungimento del 100% delle metriche soddisfatte.

4.5 Qualità del processo di verifica

4.5.1 26M-CC - Code coverage

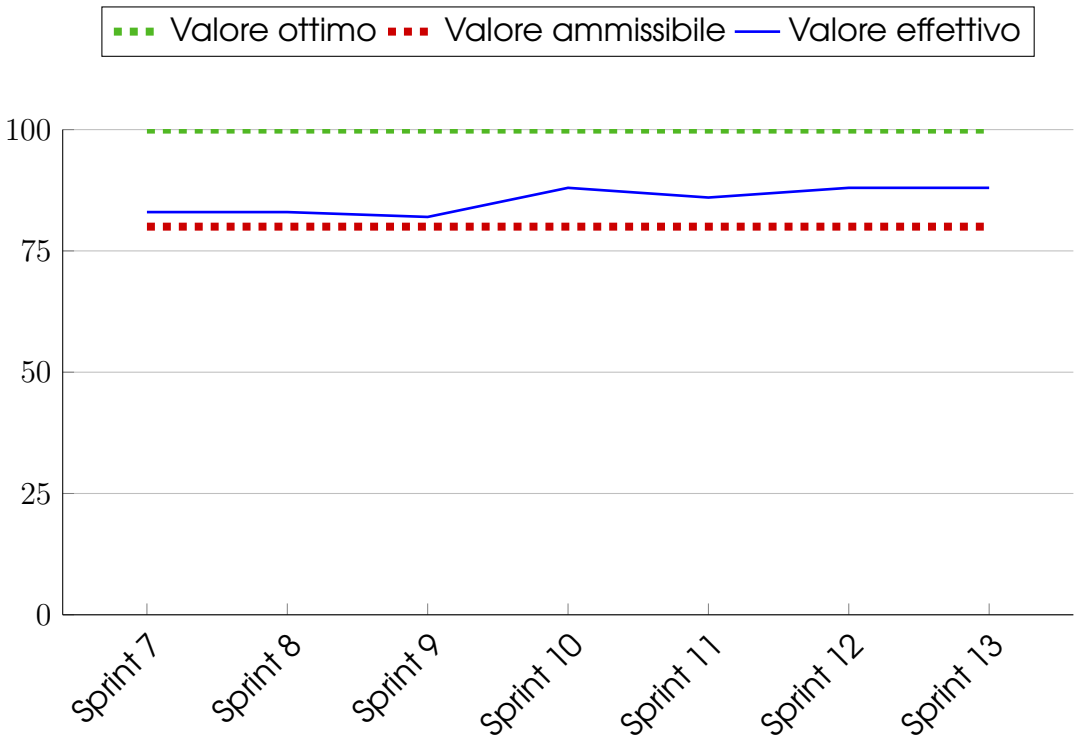

Figura 11: Percentuale di code coverage dei test implementati

PB

Al settimo sprint sono stati introdotti i primi test e fin da subito abbiamo voluto garantire la qualità del codice utilizzando le *Github Actions* per impedire che venisse effettuato il *merge* di codice non testato o che non superasse la percentuale minima prevista per ciascun test. Questo ha permesso di garantire un'alta percentuale di code coverage fin da subito, con un andamento costante e in crescita.

4.5.2 27M-BC - Branch coverage

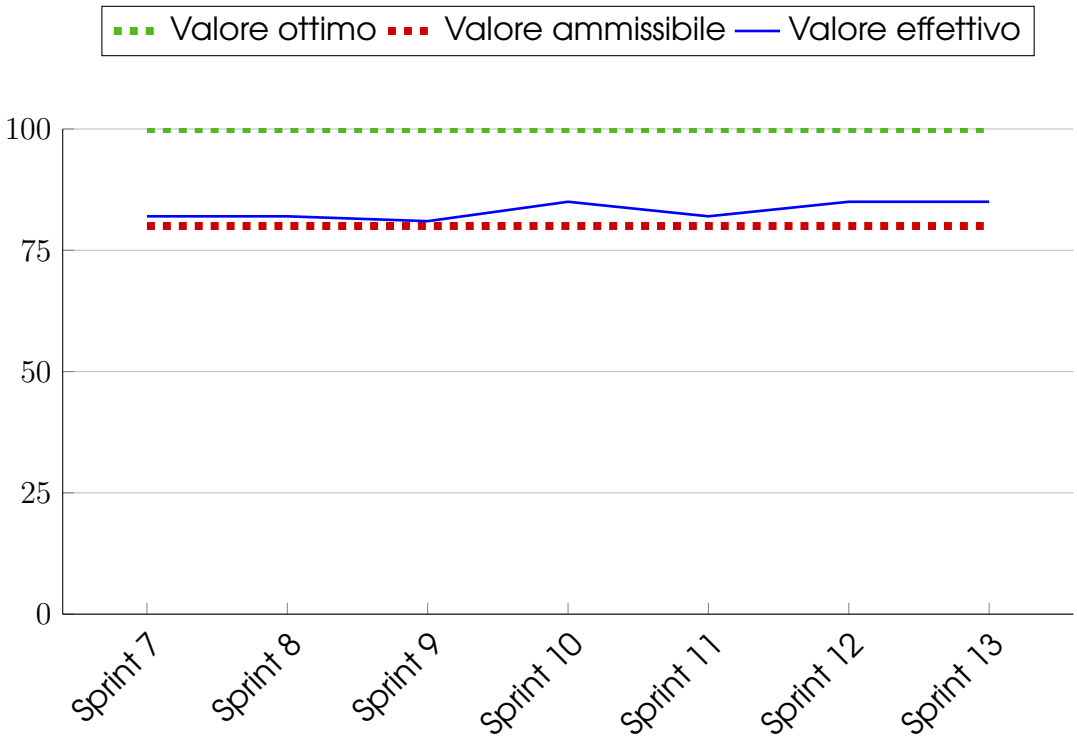

Figura 12: Percentuale di branch coverage dei test implementati

PB

Anche in questo grafico si può notare come il gruppo *7Last* abbia lavorato in modo costante per garantire la qualità del codice, con un'andamento costante e in crescita della percentuale di branch coverage. Come detto prima, questo è stato possibile grazie all'utilizzo delle *GitHub Actions* per impedire il *merge* di codice non testato o che non superasse la percentuale minima prevista per ciascun test.

4.5.3 28M-SC - Statement coverage

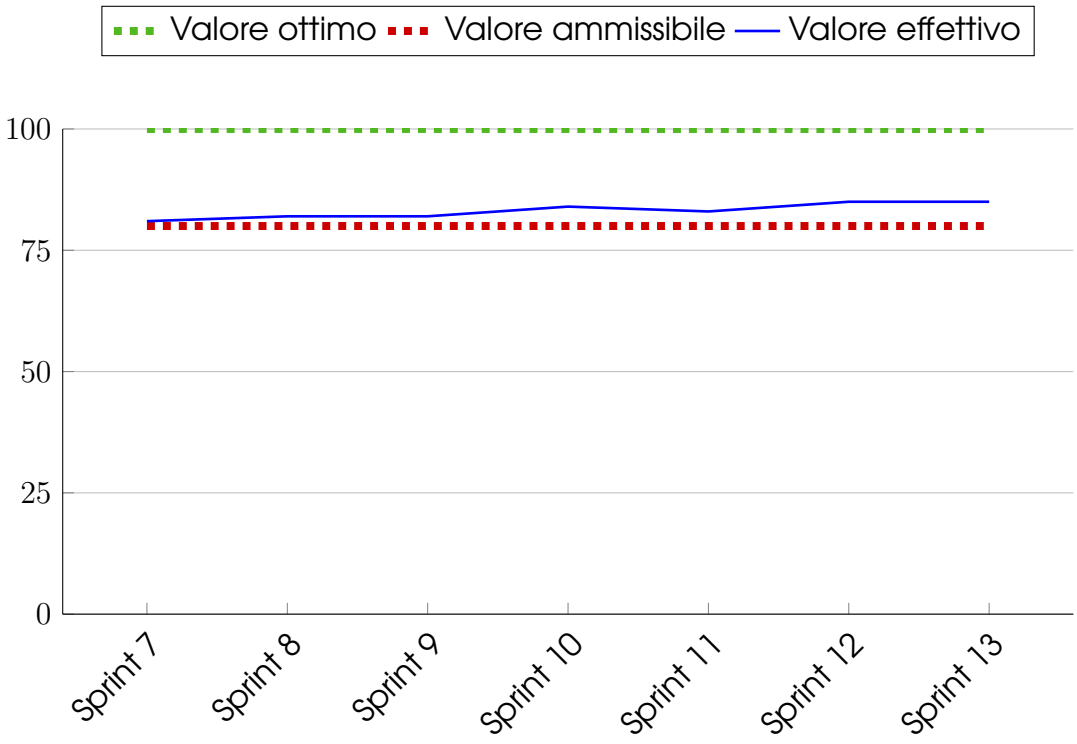

Figura 13: Percentuale di statement coverage dei test implementati

PB

Come già evidenziato in precedenza, anche qui abbiamo la conferma che il gruppo *7Last* ha lavorato per garantire la qualità del codice, con un'andamento costante e in crescita della percentuale di statement coverage. Anche in questo caso, questo è stato possibile grazie all'utilizzo delle *GitHub Actions* per impedire il *merge* di codice non testato o che non superasse la percentuale minima prevista per ciascun test.

4.5.4 29M-FD - Failure density

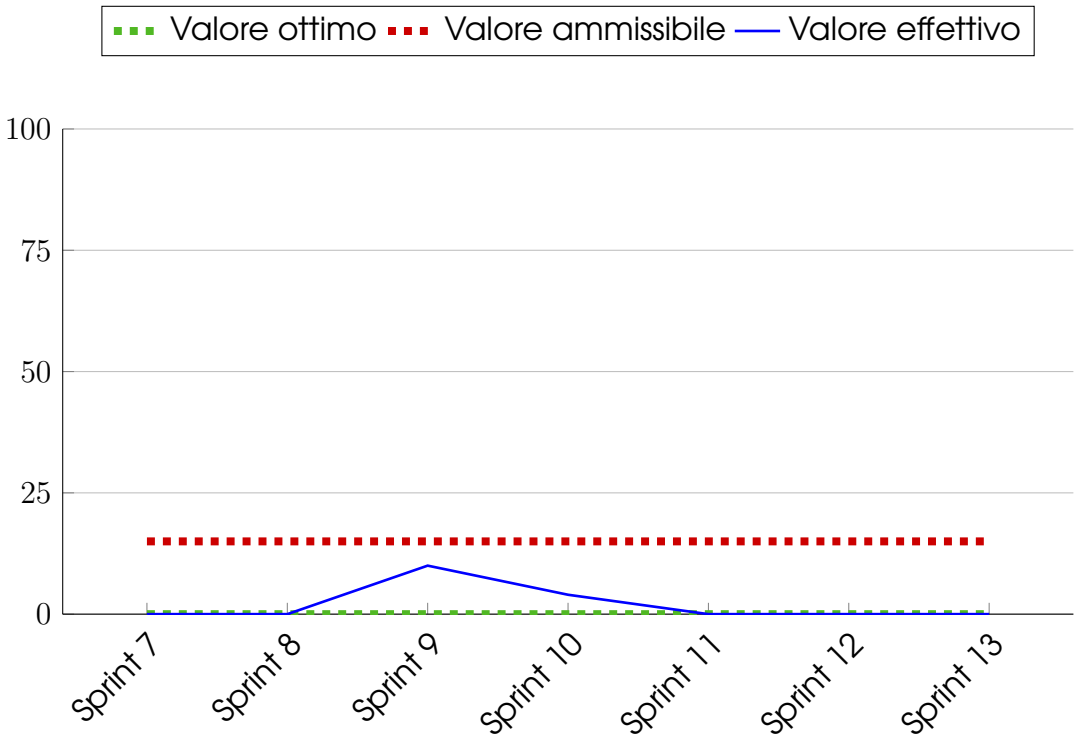

Figura 14: Percentuale di failure density

PB

Dal grafico risultano evidenti i primi fallimenti rilevati in corrispondenza del nono sprint, naturale conseguenza dell'aggiunta di nuovi test che hanno permesso di rilevare errori non precedentemente individuati e correggerli. Successivamente si nota un calo dei fallimenti fino al raggiungimento del valore ottimo.

4.5.5 30M-PTCP - Passed test case percentage

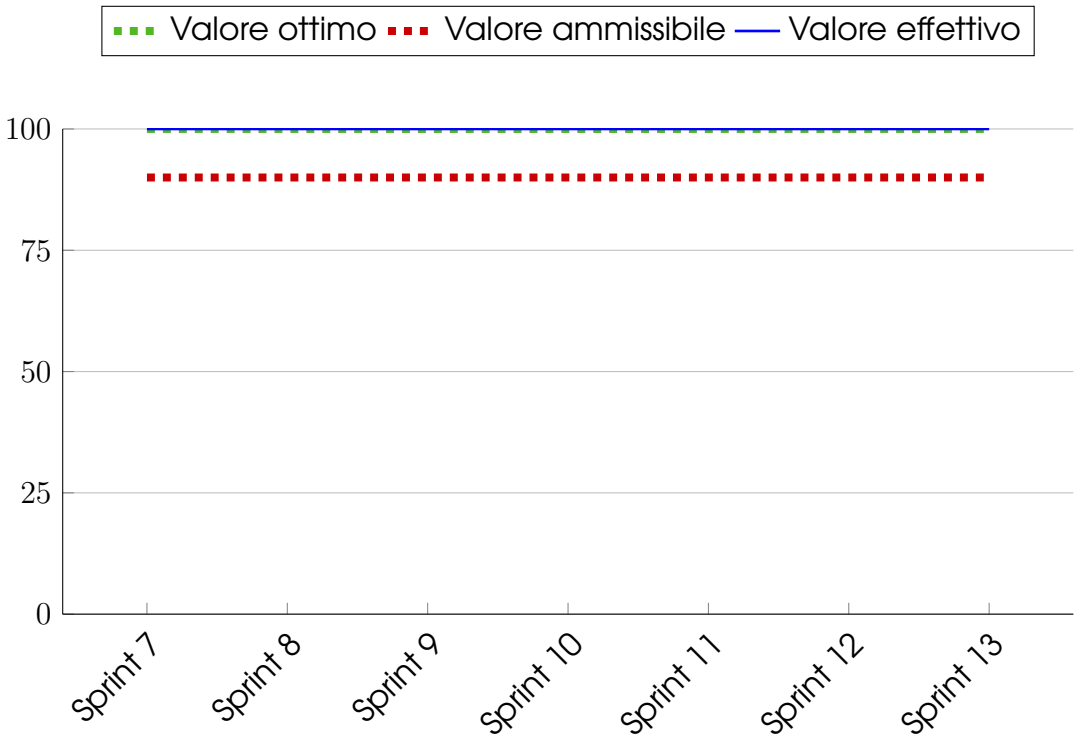

Figura 15: Percentuale di casi di test superati

PB

Grazie all'utilizzo delle *GitHub Actions*, il gruppo *7Last* è riuscito a garantire che tutti i test implementati venissero superati, con un'andamento costante fisso al 100% già dall'implementazione dei primi test in corrispondenza del settimo sprint fino alla fine del progetto.

4.6 Qualità del processo di gestione dei rischi

4.6.1 32M-NCR - Rischi non calcolati

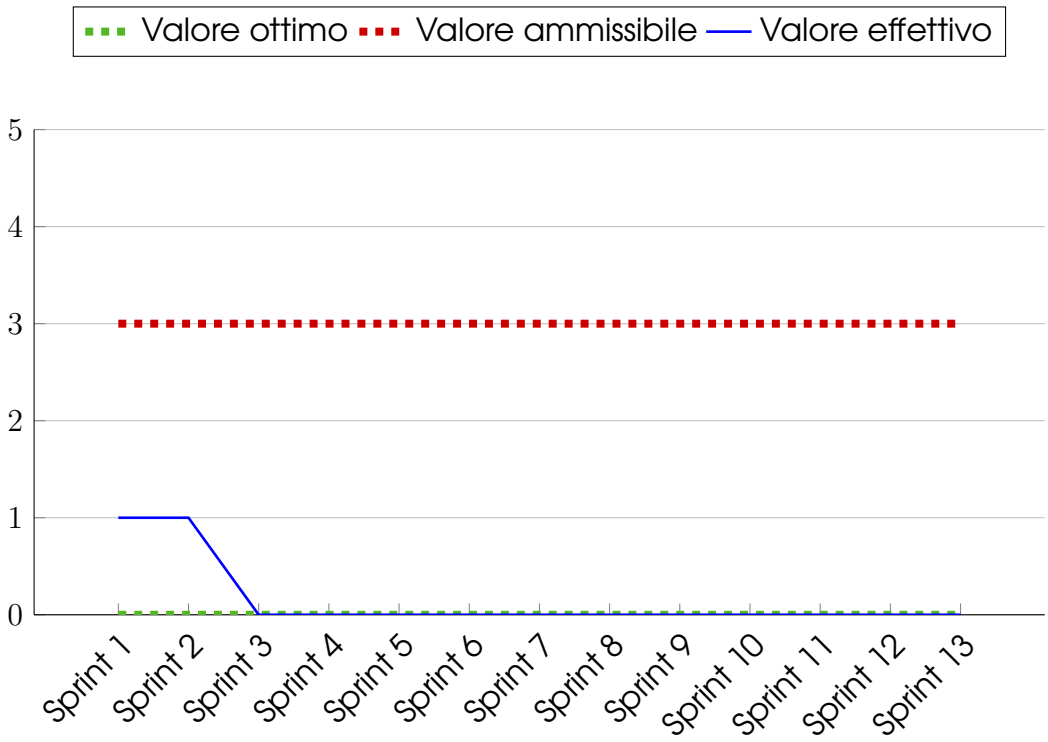

Figura 16: Rischi non calcolati occorsi durante il progetto

RTB

Dal grafico si evince che durante i primi sprint sono emersi rischi non calcolati, sintomo di una pianificazione non ottimale dovuta all'inesperienza. Successivamente il team ha accumulato esperienza, mediante automiglioramento, imparando a gestire e prevenire i rischi in modo migliore.

PB

Nella seconda parte del progetto non sono stati riscontrati rischi non calcolati, questo è dovuto all'esperienza acquisita dal gruppo *7Last* che ha permesso di prevedere e gestire in modo efficace i rischi, evitando che si verificassero.

4.7 Qualità del processo di pianificazione

4.7.1 33M-RSI - Requirements stability index

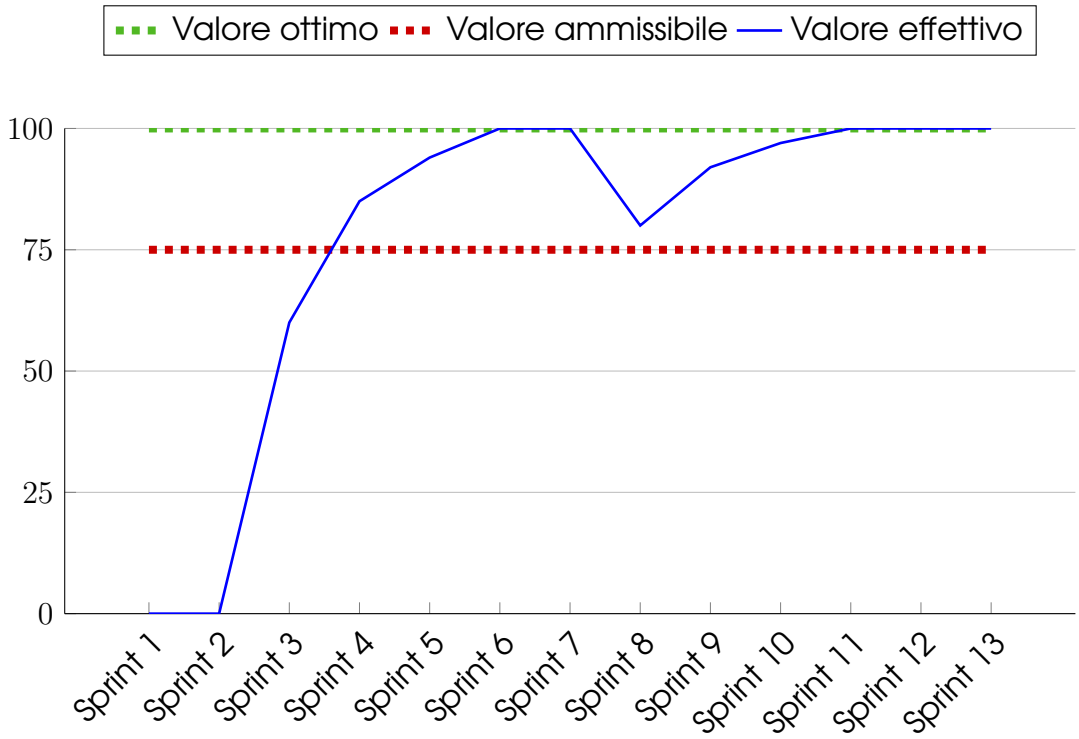

Figura 17: Percentuale di stabilità dei requisiti

RTB

L'analisi del RSI mostra un forte incremento tra il secondo e il terzo sprint, segnalando un'intensa attività di revisione e aggiustamento dei requisiti. Nei due sprint successivi, il RSI si stabilizza, indicando una riduzione delle modifiche e una maggiore stabilità dei requisiti. Questo andamento riflette un'efficace fase iniziale di consolidamento dei requisiti seguita da una stabilizzazione che facilita l'implementazione del progetto.

PB

Dal grafico si può notare come i requisiti abbiano subito alcune modifiche in corrispondenza dell'ottavo sprint (e successivi). Questo è dovuto alla necessità di apportare alcune modifiche ai requisiti in seguito ai consigli ricevuti dal professor Cardin durante la revisione RTB, oltre ad alcune variazioni concordate con il proponente su alcune funzionalità del prodotto. Questo ha permesso di migliorare la qualità del prodotto e di garantire una maggiore soddisfazione del proponente.

5 Iniziative di automiglioramento per la qualità

5.1 Introduzione

In questa sezione verranno riportate le iniziative di automiglioramento che il nostro gruppo ha deciso di adottare per aumentare la qualità del prodotto e dei processi. Queste iniziative sono state individuate grazie all'esperienza acquisita durante lo svolgimento del progetto e grazie alle valutazioni effettuate sulle attività svolte.
Trattandosi per tutti noi della prima esperienza con un progetto di questa portata, è stato necessario un grande numero di tentativi per comprendere al meglio come organizzarci e come svolgere le attività. Questo ci ha permesso di capire quali sono stati i punti di forza e i punti deboli del nostro lavoro e di individuare le aree in cui è possibile migliorare.
Per ciascuna delle difficoltà riscontrate verranno indicate:

- fase del progetto in cui si è verificato il problema;
- descrizione del problema;
- contromisura adottata per risolvere il problema evidenziato.

5.2 Problemi rilevati ed iniziative adottate

- **Organizzazione delle riunioni**
  - **Fase del progetto**: iniziale.
  - **Descrizione**: nelle prime settimane di lavoro, a partire dalla formazione dei gruppi sino ai primi Diari di bordo, si è riscontrata una certa difficoltà nell'organizzazione delle riunioni causata dai vari impegni di ciascun membro (lezioni diverse in orari diversi, lavoro per alcuni, impegni personali) e soprattutto alimentata dalle diverse riunioni che si accumulavano (SAL_G con l'azienda prima e Diari di bordo poi) portando a una certa confusione e a un rallentamento delle attività.
  - **Contromisura**: abbiamo deciso di effettuare le riunioni a distanza tramite la piattaforma *Discord* e di fissare un giorno e un orario durante la settimana per ciascuna tipologia di incontro in maniera tale da rispettare le disponibilità di ogni membro; qualora qualcuno, per impegni di natura eccezionale, non

abbia modo di essere presente potrà successivamente informarsi sui contenuti trattati attraverso i verbali che verranno redatti e messi a disposizione di tutti.

- **Suddivisione compiti**
  - **Fase del progetto**: iniziale.
  - **Descrizione**: all'inizio del progetto si è riscontrata una certa difficoltà nella suddivisione dei compiti a causa della mancanza di esperienza e della poca conoscenza delle competenze possedute da ciascuno. È risultato dunque difficile il bilanciamento delle mansioni e si sono verificati più volte casi in cui alcuni membri sono stati in grado di completare le attività a loro assegnate in anticipo, e casi opposti in cui il lavoro da svolgere è risultato eccessivo e difficilmente completabile entro i tempi prestabiliti.
  - **Contromisura**: abbiamo quindi deciso, come suggerito anche dal professor Vardanega al primo Diario di bordo, di non assegnare preventivamente tutti i compiti da svolgere a ciascun membro, ma piuttosto di metterli in un contenitore condiviso (abbiamo deciso di usare le annotazioni di *ClickUp*$_G$) e di permettere a ciascun membro di prendere in autonomia i compiti da svolgere, così che chiunque finisca in anticipo possa prenderne altri; in questo modo siamo riusciti a svolgere le attività in modo più equo e a completare i compiti entro i tempi prestabiliti.
- **Familiarità con le tecnologie**
  - **Fase del progetto**: intermedia.
  - **Descrizione**: durante lo svolgimento del progetto ci siamo resi conto che la mancanza di familiarità con le tecnologie utilizzate (in particolare con *Docker*$_G$, *Grafana*$_G$ e *Clickhouse*$_G$) ha rallentato inizialmente l'attività di sviluppo e ha portato a un aumento del carico di lavoro per alcuni membri del gruppo.
  - **Contromisura**: abbiamo deciso di organizzare un incontro di formazione in cui i membri più esperti hanno spiegato ai meno esperti il funzionamento di *Docker*$_G$ e le modalità di utilizzo. Inoltre, abbiamo deciso di utilizzare la funzionalità di *pair programming* per permettere ai membri meno esperti di lavorare a stretto contatto con quelli più esperti e di apprendere da loro.

5.3 Considerazioni finali

Fin da subito il nostro gruppo si è posto come obiettivo principale quello di dotarsi di un *Way of Working* preciso e ben definito, di pianificare ogni singola attività e di prevedere tutte le possibili difficoltà incontrabili durante lo svolgimento del progetto. Questo per cercare di prevenire i problemi e di fornire delle contromisure efficaci per affrontarli.
Inizialmente si sono presentate delle difficoltà dovute all'inesperienza del gruppo in ambito organizzativo. Tuttavia, grazie alla familiarizzazione ottenuta tramite lo svolgimento del progetto e grazie ai consigli e suggerimenti che ci sono stati forniti dai professori e dall'azienda proponente$_G$, siamo riusciti a individuare i problemi e a mettere in atto delle contromisure per risolverli.
Questo ci ha permesso di migliorare notevolmente la qualità del nostro lavoro e di svolgere le attività in modo più efficiente e più equo. Nonostante ciò siamo anche consapevoli che ci sono ancora molti aspetti su cui possiamo progredire e che ci sono ancora molte iniziative di automiglioramento che possiamo adottare. Siamo convinti che, se continueremo a lavorare con lo stesso impegno e la stessa determinazione che abbiamo dimostrato finora, saremo in grado di ottenere risultati di qualità superiore.