

PROJECT PLAN - ERB TELEMETRY

Simone Manenti 104878
Cattaneo Federico 1053265

January 1, 2024

INTRODUZIONE

1.0.1 Storia

Durante la fase di progettazione svolta dal gruppo di controllo del team ERB di formula SAE, dell'università degli studi di Bergamo, si è scoperta la necessità di dover leggere i dati dei sensori collegati alla centralina, visualizzarli a schermo per poter fare un'analisi in tempo reale e verificare che nel sistema tutti i sensori siano in uno stato accettabile. Non trovando un software gratuito che andasse in contro alle esigenze del team, abbiamo deciso di sviluppare un software con le funzionalità essenziali per poter far procedere il team con la fase di test.

Le necessità del team sono di visualizzare a schermo i dati in tempo reale, tramite tabelle o tramite grafici (funzionalità opzionale) e di salvare questi dati da qualche parte, per poter eventualmente confrontare i dati di due sessioni di test e capire se, a seguito di modifiche alla vettura, queste siano efficaci oppure siano modifiche non necessarie o controproducenti.

Descrizione del progetto

Nel sistema sono presenti due arduino:

- **arduinoTX**: collegato alla centralina, si occupa della trasmissione dei dati dalla vettura ad un ricevitore.
- **arduinoRX**: collegato ad un computer, si occupa della ricezione dei dati inviati dall'arduinoTX.

2.0.1 Funzionalità

- Riceviamo i dati in real time dei sensori collegati alla centralina nel formato: "acceleratore, pressione freno, temp. freno, temp. batteria, carica batteria, tempo sul giro, pressione gomme".
- I dati possono essere visualizzati in real time tramite tabelle o grafici.
- I dati possono essere salvati in un database locale .
- Possiamo accedere allo storico dei dati salvati nel database locale.
- L'utente che è autorizzato, può eliminare i dati non necessari, presenti nel database locale.

2.0.2 Utenti

Ci sono due possibili Livelli utente, "CapoReparto" e "IngegnereDiPista".

L'IngegnereDiPista può:

- Accedere ai dati della telemetria e visualizzarli a schermo tramite grafici o tabelle.
- Selezionare quali tipi di dati sono da visualizzare.
- Selezionare l'intervallo di tempo dei dati che vuole visualizzare.

Il CapoReparto può:

- Svolgere tutte le funzioni dell'IngegnereDiPista.
- Selezionare dei dati ed eliminarli dal database locale.

2.0.3 Comportamento comunicazione real time

- Qualsiasi utente, tramite pulsante, decide quando tentare di avviare la comunicazione tra i due arduino. Se la comunicazione è avvenuta, i dati possono essere visualizzati tramite tabella e contemporaneamente salvati in locale.
- L'utente deve anche stoppare la comunicazione tra i due arduino (implica lo stop della lettura e scrittura dei dati).
- Se la connessione tra arduino non avviene, deve apparire un messaggio di errore.

2.0.4 Comportamento dati in locale

- Qualsiasi utente può visualizzare i dati salvati nel database locale tramite tabelle o grafici.
- Qualsiasi utente può esportare i grafici o le tabelle.
- L'utente CapoReparto può selezionare ed eliminare i dati presenti nel Database.

MODELLO DI PROCESSO

Abbiamo deciso di adottare un modello di processo Agile ibrido per i seguenti motivi:

- Il team ERB attualmente non ha una visione completa del sistema
- Vogliamo procedere a piccoli incrementi per poter avere una qualità maggiore
- Potrebbero avvenire cambiamenti dell'interfaccia e delle funzioni
- Dobbiamo ottimizzare il tempo a disposizione per lo sviluppo del sistema

aggiungi tabella scrum con il punto di partenza con tutte le attività da svolgere.
aggiungi MoSCoW guardando i seguenti obbiettivi:

Dobbiamo sviluppare un software in grado di gestire i dati ricevuti.

I dati li salveremo in un database locale, in forma tabellare.

Dobbiamo avere delle funzioni per poter decidere se visualizzarli tramite tabelle o grafici.

Per gestire la modifica dei dati, introduciamo una schermata di login all'apertura del software, così da poter determinare se l'utente ha l'autorizzazione per poter eventualmente, eliminare i dati presenti nel database locale.

Dobbiamo avere una funzione che ci permetta di esportare i dati salvati nel database locale.

Dobbiamo predisporre dei popup che comunichino all'utente se la comunicazione tra il trasmettitore e il ricevitore è andata a buon fine.

ORGANIZZAZIONE DEL PROGETTO

Avendo deciso di usare un modello SCRUM, ci troviamo a dover assegnare ognuno dei tre ruoli ad entrambi i membri del Team.

I ruoli di SCRUM Master, Product owner e Development Team saranno ricoperti da:

- Cattaneo Federico
- Simone Manenti

Entrambi i membri del Team hanno pari responsabilità all'interno del progetto.

STANDARD. LINEE GUIDA E PROCEDURE

Le procedure che il team dovrà seguire sono:

- Individuazione dei requisiti, stima dei costi e delle tempistiche
- Progettazione dei modelli UML su cui si baserà il codice
- Implementazione del codice
- Verifica della qualità e della validità

Per la gestione delle versioni del sistema utilizziamo GIT, Per alcune componenti useremo uno sviluppo simile a quello adottato nell'extreme programming, nel quale entrambi i membri del Team programmeranno sulla stessa macchina.

Gestiamo la priorità dei requisiti secondo le regole MoSCoW

Adotteremo le linee guida ISO/IEC 9126

ATTIVITA' DI GESTIONE

Dopo esserci occupati dell'individuazione dei requisiti e aver fatto la prima stesura del Product Backlog, dobbiamo occuparci dello Sprint Planning.

Sprint:

- La durata di ogni sprint sarà di 7 giorni.
- Al termine del primo Sprint avverrà il rilascio di una prima versione del software, che non conterrà tutti i requisiti individuati nel Product Backlog, ma fungerà da Prototipo.
- Al termine del secondo Sprint, verrà rilasciata una versione software contenente tutti i requisiti "Must Have".
- Nei successivi Sprint andremo ad implementare i requisiti "Should have" ed eventuali nuovi requisiti "Must Have" e ci concentreremo principalmente sul miglioramento degli elementi già implementati nel software.
- Al termine di ogni Sprint presentiamo il software al team ERB per avere un feedback utile per apportare modifiche o individuare nuove funzionalità.

Daily scrum:

- I membri del team, a fine giornata, si riuniranno per pianificare l'attività del giorno dopo.

RISCHI

Abbiamo individuato diverse tipologie di rischi:

- Ritardo nel rappresentare i dati ricevuti sul TX
- Possibilità di ottenere grafici non intuitivi da leggere
- Errore di eliminazione dei dati sul database da parte del CapoReparto
- Possibilità di perdita dei dati nel caso in cui ci sia un guasto dell'hardware, poichè il database è locale, sulla macchina dell'utente

PERSONALE

Entrambi i membri del team:

- Entrambi i membri del team avranno pari responsabilità e si divideranno equamente le attività da svolgere.
- partecipano, gratuitamente, dalle prime fasi del progetto.
- hanno dovuto imparare ad utilizzare i vari software e tools proposti per lo sviluppo del sistema, ad esempio: Maven, StarUml, GitHub.
- hanno conoscenze di base del linguaggio Java.
- Non hanno competenze sulla libreria SQLite.

METODI E TECNICHE

Per lo sviluppo del progetto utilizziamo diversi software e tools:

- Eclipse e Maven
- SQLite
- GitHub
- StarUML

Per gestire la configurazione e tener traccia delle modifiche apportate al software, utilizziamo la notazione orientata alla versione;

Ogni modifica genererà una versione numerata del software (ad esempio: X.0.1).

GARANZIA DI QUALITA'

DA COMPLETARE

PACCHETTI DI LAVORO

DA COMPLETARE

RISORSE

Le funzioni svolte dal software non richiedono grandi quantità di potenza computazionale e generalmente non richiedono grandi quantità di memoria per il salvataggio dei dati.

BUDGET

Il progetto è diviso in Milestone: **Inizio del progetto:**

- L'obiettivo è di individuare il problema da risolvere, i requisiti per risolverlo e descrivere gli obiettivi.
- Concluso in data 15/12/2023

Stesura Project Plan:

- L'obiettivo è quello di generare il project plan
- Concluso in data 24/12/2023

Elaborazione dei modelli UML:

- L'obiettivo è quello di ottenere l'architettura del software tramite diagrammi UML
- Concluso in data 29/12/2023

Generazione del codice:

- L'obiettivo è quello di generare il codice a partire dai modelli UML ottenuti nella Milestone precedente ed avere le prime versioni preliminari del software
- Da concludere entro 07/01/2024

Conclusione del progetto:

- L'obiettivo è quello di verificare e validare il software e consegnare la versione definitiva al team ERB
- Da concludere entro 19/01/2024

CAMBIAMENTI

Al termine di ogni Sprint, se vengono proposti dei cambiamenti da parte del team ERB, verranno inseriti all'interno del Product Backlog per poter procedere con lo Sprint planning. Verranno anche aggiornati i documenti e verranno rilasciate le versioni del software con la numerazione descritta precedentemente.

CONSEGNA

Al termine di ogni sprint, nel caso in cui siano state sviluppate nuove versioni del software, verrà consegnata al team ERB la versione più recente e verranno descritte le funzionalità aggiunte. Ad ogni versione consegnata al team ERB, verrà rilasciata anche la documentazione inerente per poter facilitare l'utilizzo del software.

La versione finale del software verrà rilasciata solo dopo la fase di validazione e verifica fatta quando tutti i requisiti presenti nella categoria "Must Have" sono presenti all'interno del software.