Appunti sviluppo

autoauto- [Appunti sviluppo](#appunti-sviluppo)auto - [Modifiche funzionali](#modifiche-funzionali)auto - [MP/MON](#mpmon)auto - [Obiettivi](#obiettivi)auto - [Schema refactoring](#schema-refactoring)auto - [MP/MON](#mpmon-1)auto - [Obiettivi](#obiettivi-1)auto - [Schema refactoring](#schema-refactoring-1)auto - [Richieste clienti](#richieste-clienti)auto - [Jetco](#jetco)auto - [Donati](#donati)auto - [ATTENZIONE LIBRERIE](#attenzione-librerie)autoauto

Modifiche funzionali

MP/MON

Obiettivi

Modifica comportamento MP/MON:

- rendere la pagina del MON configurabile per ogni cliente
- migiorare il carico sul DB sfruttando meglio la cache di redis
- costruire uno strato intermedio di configurazione x la visualizzazione che permetta di disaccoppiare visualizzazione cliente e recupero dati (più standard)
- avere una policy di recupero e calcolo dati da DB suddivisa per NATURA del dato in modo da eseguire meno di frequeste i task meno importanti

Schema refactoring

Si prevede di effettuare il refactoring seguendo queste linee guida:

- uso intensivo di LUT (LookUp Table)
- definire un area PER OGNI IMPIANTO in REDIS
- definire gli elementi di ogni singolo blocco come posizioni in cui mostreremo una LABEL
- in ogni area redis avremo TUTTE le informazioni di una macchina
- ci potranno essere più HashTable / KVP per ogni macchina, per rappresentare ad es diverse sorgenti di dato rappresentabili con un HAS REDIS per venire recuperate
- creeremo una LUT chiamata **LUT_TTL** che configurerà il TTL standard per ogni BLOCCO informativo (ad esempio: PROD1 --> 2 significa che il dato di PROD1 avrà un TTL di 2 sec dopo di cui sarà ricalcolato); TTL = 0 --> nessuna scadenza, altrimenti si aspetta un TTL in sec (decimal per avere anche < 1 sec)
- associeremo una **LUT_DATA** x legare le labels posizionali agli elementi delle aree informative in REDIS (es: label1 --> PROD:CodArt)
- chiamate tramite API REST
- cercheremo per prima cosa ogni items in REDIS, non trovandolo sarà invocato il metodo di refresh della HashTable / sorgente informativa con il TTL definito nella **LUT_TTL**
- compost il dato complessivo questo sarà salvato in una cache ELABORATA di breve periodo (tipicamente <= 500ms) in modo da non rifare nemmeno letture/ricostruzioni oltre i 2Hz
- tendenzialmente ogni singolo blocco CHIAMERA' la WebApi x ottenere le proprie informazioni e mostrarle

- le LUT saranno configurate nel DB in modo differente per ogni cliente, avendo una versione standard sulla macchina **STD_IOB** (in modo da poterla usare come default ove non specificato) e avendo poi override per ogni macchina in cui fosse necessario per le sole parti differenti. Ad esempio, se ogni macchina si differenziasse solo per le etichette label3 e label4, avremo il set completo della **STD_IOB**, poi per ogni impianto avremo la definizione di come comporre laberl3/label4 sul DB; a livello REDIS questi dati saranno "fusi" per costituire le essatte LUT di ogni macchina in modo esplicito (e teoricamente immutabile sino a riavvio dei pool di esecuzione delle applicazioni /MP/MON)

DI FATTO possiamo usare delle PARTIAL PAGE (come le attuali _StatusMapSingle.cshtml / _StatusMap.cshtml) x definire il MACRO template.

Ogni machcina specificherà QUALE template usare (SE NON QUELLO STANDARD, ereditato)

Ogni tempalte userà dei dati in modo diverso a partire dal datamodel COMUNE ED UNICO che espone per TUTTE le macchine lo stesos insieme COMPLETO di dati.

MP/MON

Obiettivi

Modifica comportamento MP/MON:

- rendere la pagina del MON configurabile per ogni cliente
- migiorare il carico sul DB sfruttando meglio la cache di redis
- costruire uno strato intermedio di configurazione x la visualizzazione che permetta di disaccoppiare visualizzazione cliente e recupero dati (più standard)
- avere una policy di recupero e calcolo dati da DB suddivisa per NATURA del dato in modo da eseguire meno di frequeste i task meno importanti

Schema refactoring

Si prevede di effettuare il refactoring seguendo queste linee guida:

- uso intensivo di LUT (LookUp Table)
- definire un area PER OGNI IMPIANTO in REDIS
- definire gli elementi di ogni singolo blocco come posizioni in cui mostreremo una LABEL
- in ogni area redis avremo TUTTE le informazioni di una macchina
- ci potranno essere più HashTable / KVP per ogni macchina, per rappresentare ad es diverse sorgenti di dato rappresentabili con un HAS REDIS per venire recuperate
- creeremo una LUT chiamata **LUT_TTL** che configurerà il TTL standard per ogni BLOCCO informativo (ad esempio: PROD1 --> 2 significa che il dato di PROD1 avrà un TTL di 2 sec dopo di cui sarà ricalcolato); TTL = 0 --> nessuna scadenza, altrimenti si aspetta un TTL in sec (decimal per avere anche < 1 sec)
- associeremo una **LUT_DATA** x legare le labels posizionali agli elementi delle aree informative in REDIS (es: label1 --> PROD:CodArt)
- chiamate tramite API REST
- cercheremo per prima cosa ogni items in REDIS, non trovandolo sarà invocato il metodo di refresh della HashTable / sorgente informativa con il TTL definito nella **LUT_TTL**
- compost il dato complessivo questo sarà salvato in una cache ELABORATA di breve periodo (tipicamente <= 500ms) in modo da non rifare nemmeno letture/ricostruzioni oltre i 2Hz

- tendenzialmente ogni singolo blocco CHIAMERA' la WebApi x ottenere le proprie informazioni e mostrarle
- le LUT saranno configurate nel DB in modo differente per ogni cliente, avendo una versione standard sulla macchina **STD_IOB** (in modo da poterla usare come default ove non specificato) e avendo poi override per ogni macchina in cui fosse necessario per le sole parti differenti. Ad esempio, se ogni macchina si differenziasse solo per le etichette label3 e label4, avremo il set completo della **STD_IOB**, poi per ogni impianto avremo la definizione di come comporre laberl3/label4 sul DB; a livello REDIS questi dati saranno "fusi" per costituire le essatte LUT di ogni macchina in modo esplicito (e teoricamente immutabile sino a riavvio dei pool di esecuzione delle applicazioni /MP/MON)

DI FATTO possiamo usare delle PARTIAL PAGE (come le attuali _StatusMapSingle.cshtml / _StatusMap.cshtml) x definire il MACRO template.

Ogni machcina specificherà QUALE template usare (SE NON QUELLO STANDARD, ereditato)

Ogni tempalte userà dei dati in modo diverso a partire dal datamodel COMUNE ED UNICO che espone per TUTTE le macchine lo stesos insieme COMPLETO di dati.

Richieste clienti

Jetco

- da completare traduzione prog CNC di GD32 e GLD25
- se possibile leggere (fanuc/siemens) i PEZZI LANCIATI che sono <= dei pezzi richiesti dall'Ordine di Produzione (es ordine: 20'000, lancio 5'000, poi 10'000, poi 15'000, poi 21'000 che finisce la notte)

Donati

- report produzione
- fattura contratto 1° sem 2018
- Install EMCO
- install Scatole (x4, 2 robotservice, 2 altri)
- install combi ROBOT (D)
- install doosan (nuove, x2)

ATTENZIONE LIBRERIE

Redis e Reis.StrongName vanno tenuti a vers 1.2.1:

- https://github.com/StackExchange/StackExchange.Redis/issues/853
- https://github.com/StackExchange/StackExchange.Redis/issues/871

inoltre microsoft.web.redisSessionstateProvider va tenuto alla 2.2.6

- https://github.com/Azure/aspnet-redis-providers/wiki