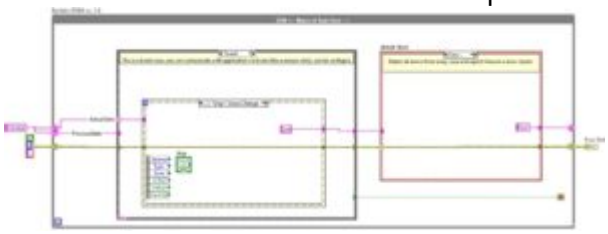
Design Pattern 3# Event-Based State Machine

Category: LabVIEW Talk
scritto da Davide Oratelli | Marzo 18, 2020



Event-Based State Machine

Nel precedente articolo abbiamo parlato della Simple State Machine e del caso Idle, accennando alla Event Structure.

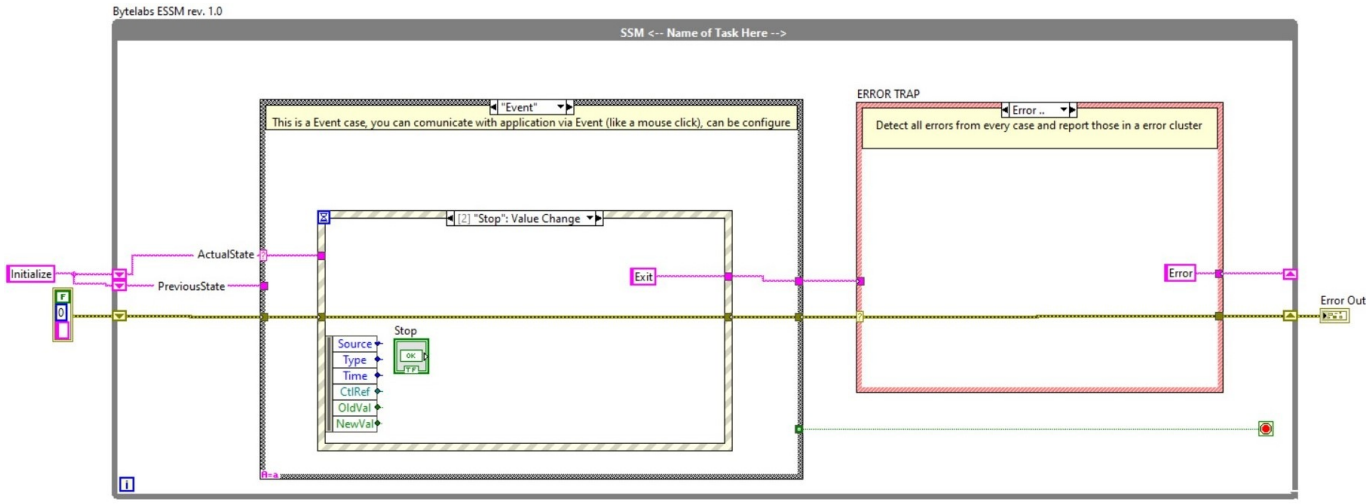
L'Event-Based State Machine combina la possibilità di interagire con l'utente alla macchina a stati precedente.

La particolarità di questo pattern è proprio la possibilità di creare un applicazione che intergisca completamente con l'utente.

Ogni evento che si manifesta all' interno della macchina a stati, porta ad un certo case che esegue codice in base alla scelta che l'utente compie all'interno dell'applicazione.

Quando viene eseguita in LabVIEW, il VI viene sospeso fino a quando si verifica uno degli eventi per i quali è configurato per l'ascolto, attendendo quindi che l'utente influisca nel programma.

L'event-Based State Machine si presenta con questo pattern:



Come la Simple State Machine, anche nell'Event Based SSM vi è un while loop e una case structure e anche qui gli shift register hanno il compito di memorizzare lo stato corrente all' iterazione.

Nel programma si può notare un case structure nominato ‘ERROR TRAP‘, che ha il compito di catturre gli errori che non possono essere ignorati.

Con l’aggiunta dell’ Event structure adesso l’applicazione è quindi in grado di interagire con l’utente, con lo scatenarsi di un evento allo stato desiderato.

Esempio: se vogliamo uscire dalla nostra applicazione vi sarà un’ evento di uscita che se scatenato porterà all’ uscita dal programma tramite un case.

Esempio: se vogliamo stoppare il programma, basterà cliccare il pulsante stop (se esso ha un evento associato) per portarci nel case nominato ‘stop’, che eseguirà quindi il codice legato a quel caso per fermare l’applicazione.

L’integrazione della Event Structure nella Macchina a stati risulta quindi ideale per gestire le UI eliminando polling inutili e dispendiosi.

Nel prossimo articolo vedremo la funzionalità della Multiple State Machine