

Gestione impianto di pompaggio con rotazione dei motori

Questo impianto dimostrativo esemplifica come sia possibile ottenere utilizzando il web-engine e we-devtool uno schema di controllo piuttosto complesso per soddisfare anche gli utenti più smaliziati.

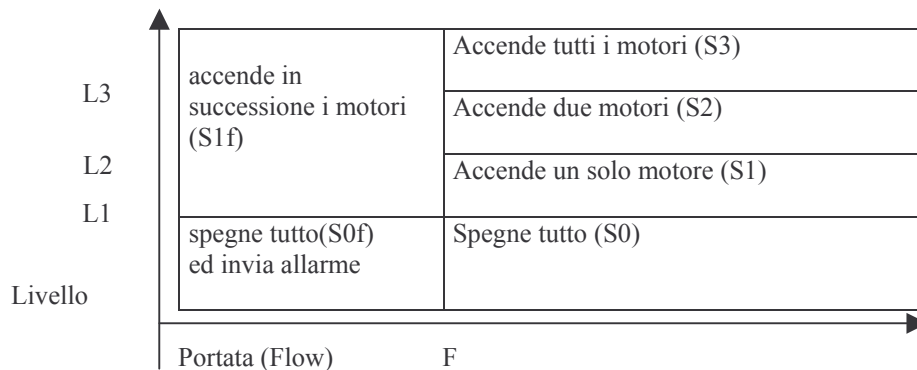
Supponiamo di avere a disposizione un impianto di pompaggio con tre pompe P1,P2 e P3 che si occupano di svuotare un serbatoio.

Supponiamo inoltre di avere un sensore di livello del serbatoio dell'acqua ed un sensore di portata.

Il comportamento desiderato è quello di evitare che il serbatoio si riempia troppo e trabocchi, così come quello di garantire una sufficiente portata.

Ovviamente poi si desidera fare in modo che i motori vengano utilizzati uniformemente, stabilendo una rotazione data dal tempo di utilizzo (piccole modifiche possono far sì che a governare il comportamento siano invece il numero di rispettive accensioni.)

Una schematizzazione del funzionamento può essere la seguente(in ascissa la portata, in ordinata il livello):



L1,L2 ed L3 sono tre soglie di livello configurabili. F è la soglia di portata.

Ovviamente la scelta del motore da accendere (o da spegnere) deve avvenire in base all'effettivo utilizzo. A tale scopo sono definite le variabili min1,min2,min3 che vengono aggiornate ogni minuto di funzionamento della pompa corrispondente¹.

Costruzione dell'impianto

La prima operazione da effettuare è quella di definire le variabili utilizzate:

P1: pompa 1(digital out 1)

P2: pompa 2(digital out 2)

P3: pompa 3(digital out 3)

Level: livello acqua (analog in 1)

Flow: portata (analog in 2)

L1: soglia superata la quale solo un motore è acceso

L2: soglia superata la quale due motori sono accesi

L3:soglia superata la quale i tre motori sono tutti accesi

L4:soglia di guardia: al superamento viene inviato un sms di allarme

¹ Non è stato previsto alcun meccanismo di ripristino qualora una di queste variabili cresca troppo e causi un overflow, in quanto prima di tale avvenimento devono passare circa 4000 anni.

F: soglia di portata

Min1: minuti utilizzo motore1

Min2: minuti utilizzo motore2

Min3: minuti utilizzo motore3

In secondo luogo si devono definire delle condizioni in maniera da definire sei stati (uno per ogni area del diagramma)

l1: livello>L1

l2: livello>L2

l3: livello>L3

l4: livello>L4

ll1: livello<L1

ll2: livello<L2

ll3: livello<L3

ll4: livello<L4

p1on: P1=accesa

p2on: P2=accesa

p3on: P3=accesa

f0: Flow<F

f1: Flow>F

min12: Min1<Min2

min13: Min1<Min3

min23: Min1<Min3

min123: Min1<Min2 AND Min1<Min3

Lo stato S1 ad esempio è definito da livello>L1 AND Livello<L2 AND Flow>F (cioè l1, ll2 e f1)

Schematizzazione delle azioni

L'idea alla base delle azioni è piuttosto semplice: per ogni stato creo un evento. In corrispondenza di quest'evento definisco che motori dovranno essere accesi.

Questo è semplice per gli stati S0, S0f ed S3. Risulta un po' più complicato per S1 ed S2.

In pratica

Quando mi trovo in S0 o S0f	spengo tutto
Quando sono in S3	accendo tutto
Quando sono in S1	faccio in modo che sia acceso un solo motore(quello meno usato)
Quando sono in S2	Faccio in modo che siano accesi due motori (quelli meno usati)
Quando sono in S1f	Accendo in successione i motori finché la portata non risale o tutti e tre sono accesi o svuoto il serbatoio (nel qual caso si ricade nel caso S0f)

La parte problematica è quella di fare in modo che in S1 ed S2 vengano attivati i motori meno usati tra quelli che sono spenti (o viceversa vengano spenti i motori più utilizzati tra quelli accesi)

Qui vengono in aiuto le strutture IF-THEN-ELSE, dette anche alberi decisionali.

In pratica nella agli eventi che individuano gli stati S1 ed S2 si associa un albero decisionale che viene eseguito ogni 10 secondi fintanto che si permane nello stato. La struttura viene eseguita nella loop action e non nella start o stop action per garantire che qualunque sia la transizione che porta allo stato di arrivo il comportamento finale sia corretto.

Siccome la logica di controllo espressa in termini di espressioni condizionali (IF...THEN..ELSE) è piuttosto complessa, si dà di seguito una schematizzazione degli alberi di decisione

Per garantire inoltre la rotazione delle pompe, dopo che il sistema permane nello stato S1 o S2 per troppo a lungo, viene forzato lo spegnimento della pompa più utilizzata. A questo punto il sistema si accorgerà entro i 10 secondi successivi che il numero di pompe accese non corrisponde a quello desiderato, e provvederà ad accendere la pompa meno utilizzata tra quelle spente.

Schematizzazione degli alberi di decisione

Vengono definiti quattro alberi decisionali:

Activate1: fa in modo che ci sia attivato un solo motore
 Activate2: fa in modo che ci siano attivati due motori
 Accendiuna: accende una pompa (la meno usata tra quelle spente)
 Spegniuna: spegne la pompa più usata tra quelle accese

A seconda degli stati assunti dalle tre pompe (acceso o spento) si vengono a creare otto casi possibili come si vede dalla tabella seguente.

Activate1 (attiva un solo motore)

P1	P2	P3	azione
ON	ON	ON	Spegni una tra 1 e 2 ⁽¹⁾
		OFF	Spegni una tra 1 e 2
	OFF	ON	Spegni una tra 1 e 3
		OFF	OK
OFF	ON	ON	Spegni una tra 2 e 3
		OFF	OK
	OFF	ON	OK
		OFF	Accendi una tra 1, 2 e 3

1- siccome l'azione viene valutata ogni 10 secondi la seconda verrà spenta poco dopo

Activate2 (attiva due uscite)

P1	P2	P3	azione
ON	ON	ON	Spegni una tra 1, 2 e 3
		OFF	OK
	OFF	ON	OK
		OFF	Accende una tra 2 e 3
OFF	ON	ON	OK
		OFF	Accende una tra 1 e 3
	OFF	ON	Accende una tra 1 e 2
		OFF	Accende una tra 1 e 2 ⁽²⁾

2- siccome l'azione viene valutata ogni 10 secondi la seconda verrà accesa poco dopo

La descrizione degli alberi decisionali Accendiuna e Spegniuna non viene qui riportata.

La dicitura "Accendi una tra 1 e 2" sta a significare che dovrà essere accesa la pompa con minor utilizzo tra la 1 e la 2. Questo è facilmente ottenibile con un'ulteriore livello di annidamento dell'IF.

Appendice: codice C corrispondente all'albero "Activate1"

Il codice seguente mostra in linguaggio C le azioni che l'albero di decisione "Activate1" esegue.
Nota: min12 significa $Min1 < Min2$. Si è scelto qui di mantenere la convenzione usata dal programma.

```
if(p1==on)
{
    if(p2==on)
    {
        if(min12)
            spegni(2);
        else
            spegni(1);
    }
    else
    {
        if(p3==on)
        {
            if(min13)
                spegni(3);
            else
                spegni(1);
        }
        else
        {
        }
    }
}
else
{
    if(p2==on)
    {
        if(p3==on)
        {
            if(min23)
                spegni(3);
            else
                spegni(2);
        }
        else
        {
        }
    }
    else
    {
        if(p3==on)
        {
        }
        else
        {
            if(min123)
            {
                accendi(1);
            }
            else
            {
                if(min23)
                    accendi(2);
                else
                    accendi(3);
            }
        }
    }
}
}
```