Universitatea ”Petru Maior” Târgu Mureș
Facultatea de Inginerie
Specializarea: Managementul Sistemelor de Energie

Referat Nr. 2

Programarea aplicațiilor SCADA - Utilizarea limbajului Cicode

Masterand:
FEKETE Albert-Zsombor

2012

1. Obiective

- Utilizarea limbajului de programare Cicode in cadrul unui proiect SCADA.
- Utilizarea instructiunilor decizionale si a functiilor definite de utilizator pentru implementarea functionalitatii unei pagini grafice.
- Realizarea unei pagini grafice in care se folosesc instructiuni de atribuire si instructiuni decizionale.
- Realizarea unei pagini grafice in care se folosesc instructiuni repetitive.

2. Realizarea practică a aplicațiilor

Primul pas în realizarea aplicațiilor propuse a fost descărcarea și deschiderea (prin funcția restore) proiectului de bază: *Sch_el_Start*. Prin elementul de meniu File->New Page am adăugat o nouă pagină grafică, pe care l-am salvat cu numele: *labs2_01* (*fig. 1*).

Funcții realizate (*fig. 1*):

- Comutatorul poate fi comandat în mai multe feluri: cu ajutorul butoanelor Pornire și Oprire, prin butonul Pornire/Oprire utilizând funcția *toggle()* sau cu ajutorul set simbolului situat în schemă.
- Textul alocat ledului care indică starea sistemului este schimbat în funcția stării.

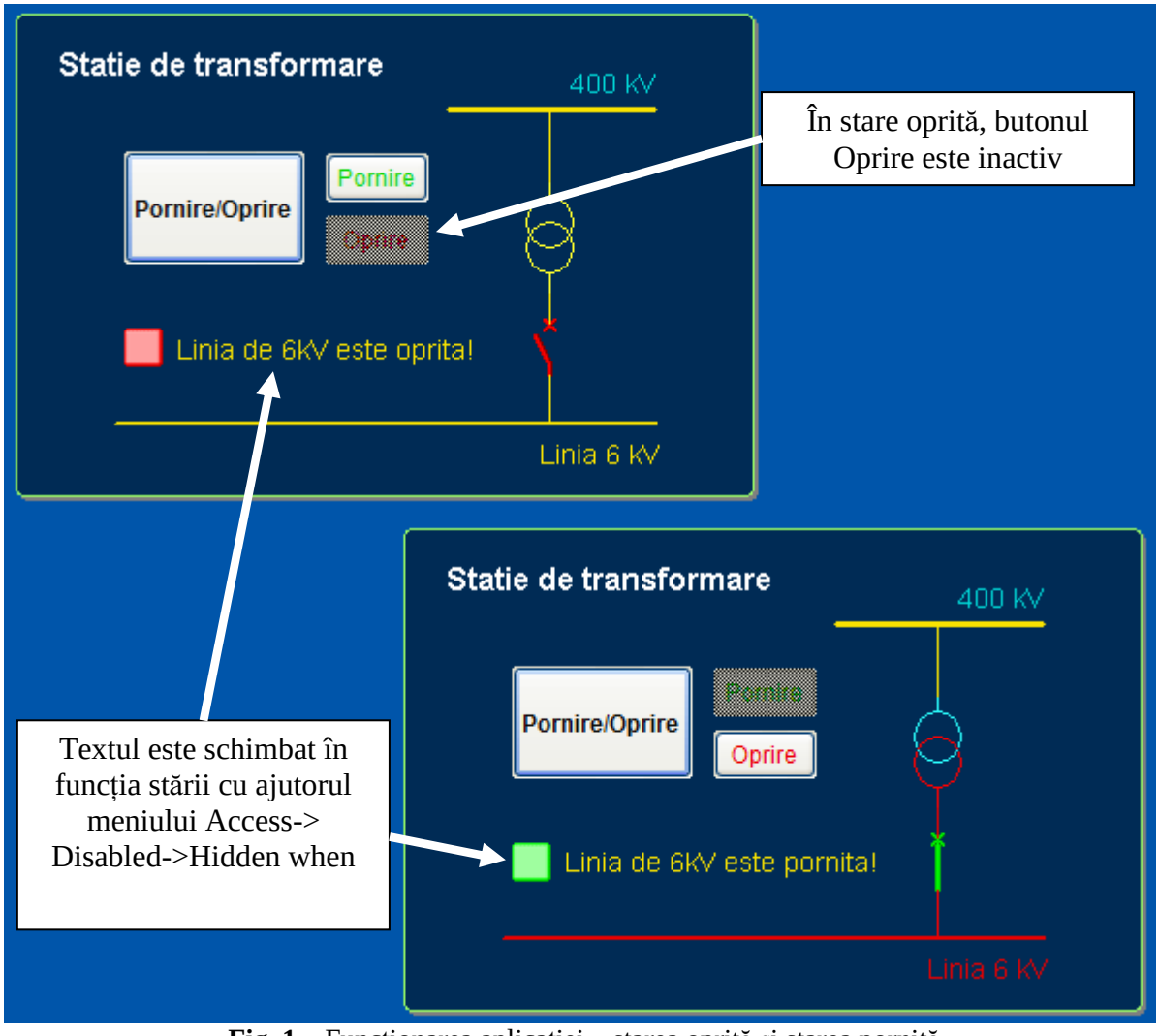

Fig. 1 – Funcționarea aplicației – starea oprită și starea pornită

Schimbarea textelor în funcția stării este realizat în felul următor. Sunt utilizate două elemente grafice *Text* care cu ajutorul meniului *Arrange* sunt suprapuse. În ambele cazuri este activat opțiunea Access->Disabled->Hidden when (*fig. 2*).

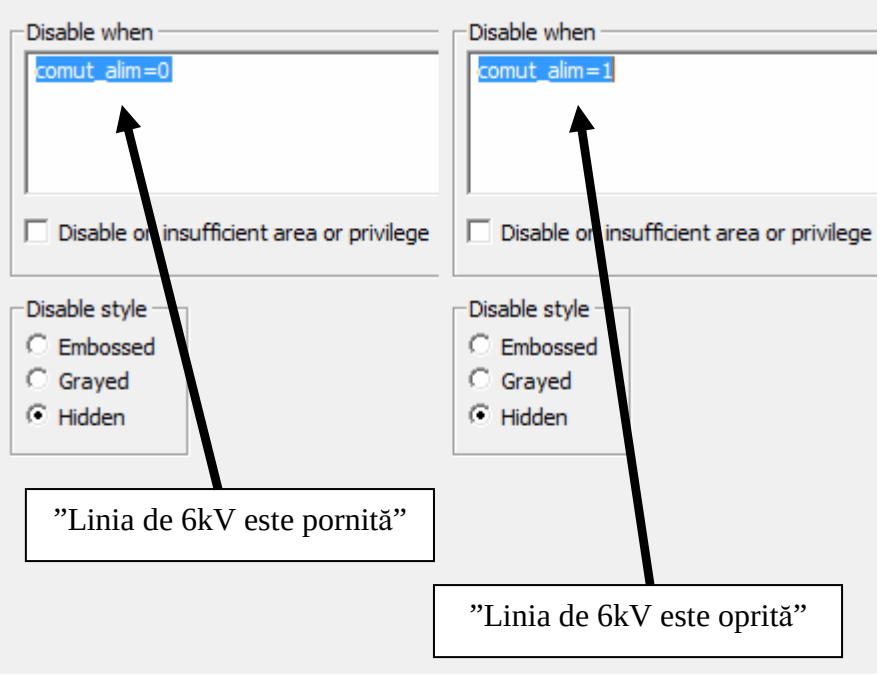

Fig. 2 – Schimbarea textelor în funcția stării comutatorului

Am introdus un singur tag din meniul Local Variables de tip Digital. Structura și configurația acestuia se poate vedea în *tabelul 1*, respectiv în *fig. 3*.

Nr.	Tag	Tip	Notă
1	comut_alim	Digital	Stare comutator alimentare

Tabelul 1. – Configurația tag-ului local

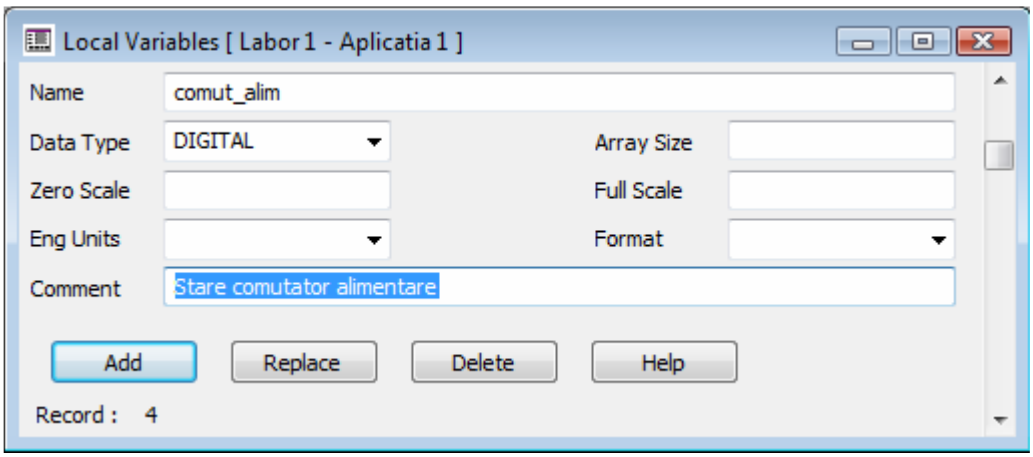

Fig. 3 – Configurația tag-ului local

În cea de a doua aplicație am utilizat două tag-uri de tip vector (*fig. 4*). Primul este un vector cu elemente de tip INT pentru valorile cuprinse între 0 și 100. Al doilea este un vector cu elemente de tip DIGITAL pentru cele 10 leduri. Aceste leduri sunt asociate cu cele 10

elemente grafice pentru incrementarea și decrementarea valorilor de tip INT menționat mai înainte.

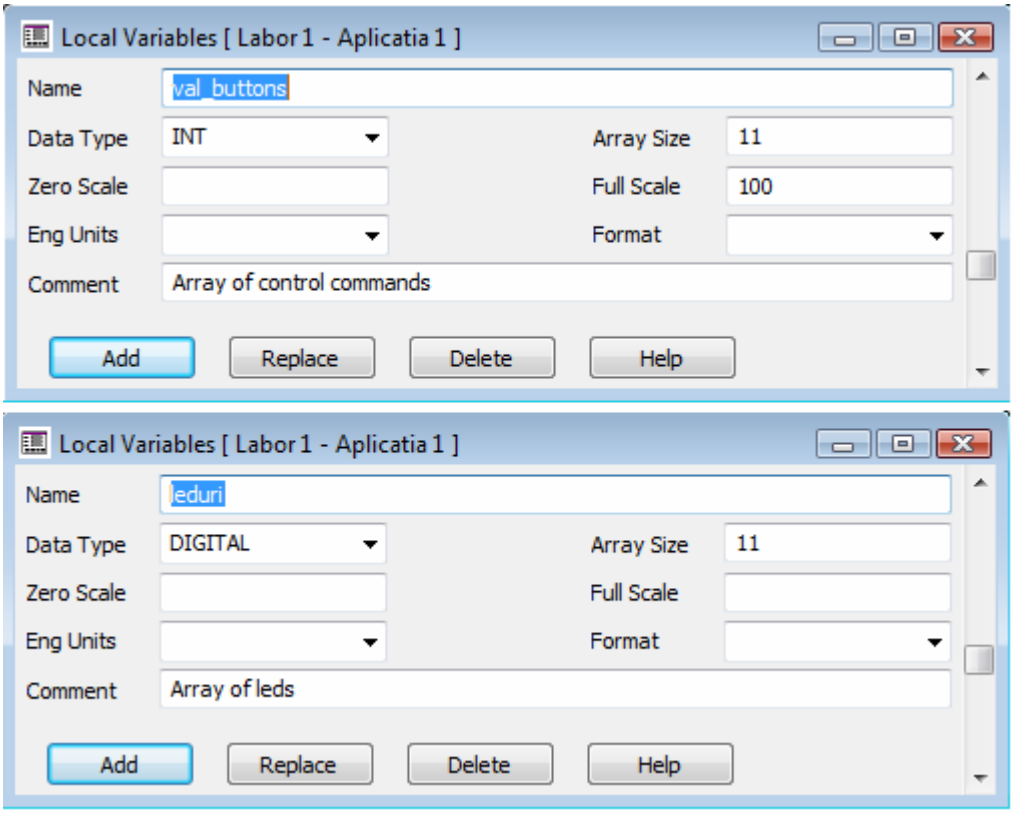

Fig. 4 – Configurația tag-urilor de tip vector

Funcții realizate:

- Patru butoane de comandă: Start, Stop, Rand pentru generarea valorilor în mod random, Init pentru inițializarea valorilor cu 0.
- La apăsarea fiecărui buton se rulează o funcție proprie.
- Este folosit un led pentru indicarea stării sistemului. La realizarea acestuia s-a folosit un Set Simbol de tip ***Multi-State***.
- Sunt folosite patru texte diferite pentru cele patru stări: Ledurile sunt stinse, Ledurile sunt aprinse, Proces de aprindere, Proces de stingere. Realizarea schimbării textelor este bazat pe metodologia prezentată în pagina 2.

Hidden when:

”Ledurile sunt stinse”
lp1[1]=1 or(lp1[1]=0 and lp1[2]=1)
”Proces de aprindere”
(lp1[1]=0 and lp1[2]=1)or(lp1[1]=0 and lp1[2]=0)or(lp1[1]=1 and lp1[2]=1)
”Ledurile sunt aprinse”
(lp1[1]=1) or(lp1[1]=1 and lp1[2]=0)or(lp1[1]=0 and lp1[2]=0)
”Proces de stingere”
(lp1[1]=0 and lp1[2]=1)or(lp1[1]=0 and lp1[2]=0)or(lp1[1]=1 and lp1[2]=0)

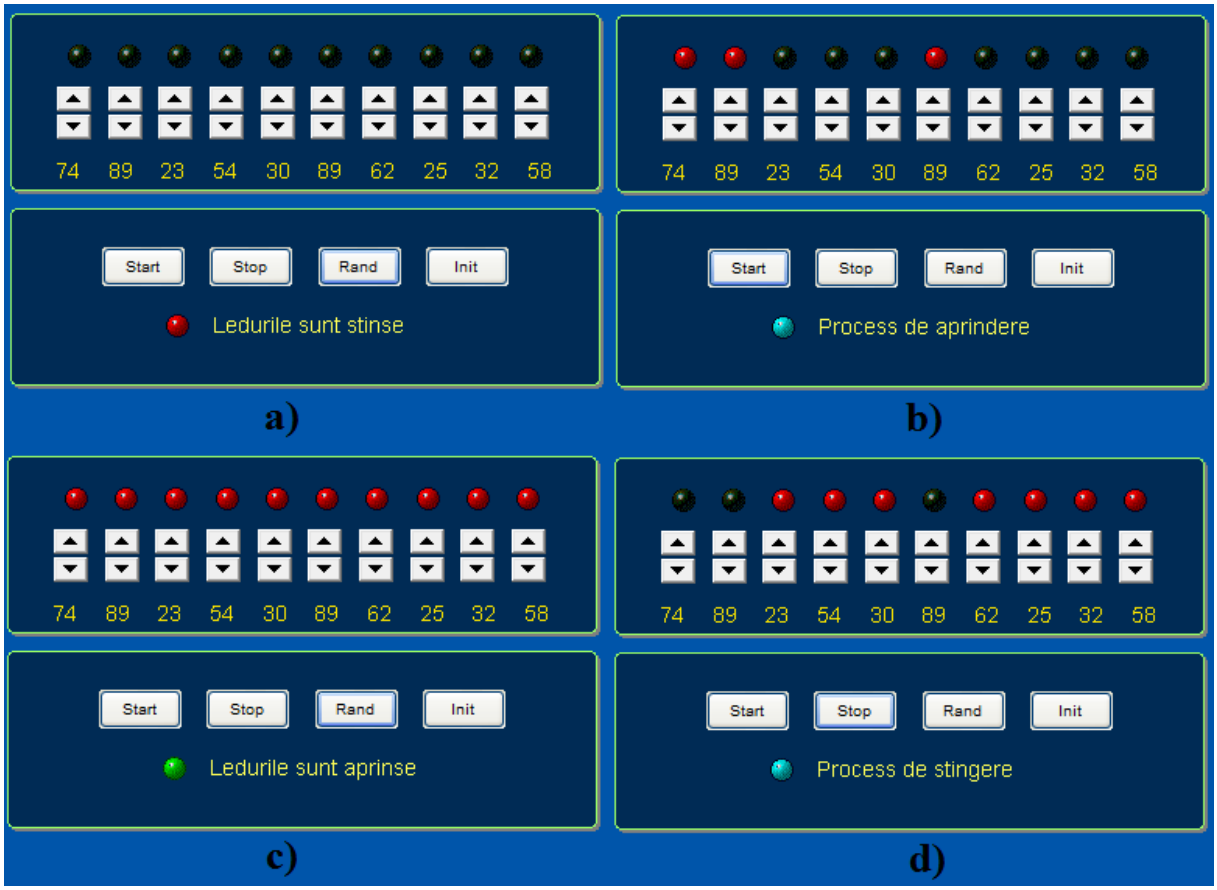

Fig. 5 – a). Ledurile sunt stinse, a fost apăsat butonul Rand b). Proces de aprindere, a fost apăsat butonul Start c). Ledurile sunt aprinse d). Proces de stingere, a fost apăsat butonul Stop

Funcțiile realizate:

```
FUNCTION Random_number()
INT i;

FOR i=1 TO 10 DO
    val_buttons[i]=Rand(100);
END
END

FUNCTION init_number()
INT i;

FOR i=1 TO 10 DO
    val_buttons[i]=0;
END
END

FUNCTION ecran_lab2()
INT i;

FOR i=1 TO 10 DO
    IF val_buttons[i]<0 THEN
       val_buttons[i]=0;
    END
END
```

```
FUNCTION start_leds()
INT i;
INT j;
INT mx_val;
INT poz;

lp1[1]=1;
lp1[2]=0;

FOR i=1 TO 10 DO
    mx_val=0;
    FOR j=1 TO 10 DO
        IF val_buttons[j]>=mx_val AND leduri[j]=0 THEN
           mx_val=val_buttons[j];
           poz=j;
        END
    END
    leduri[poz]=1;
    Sleep(1);
END

lp1[1]=0;
lp1[2]=1;

END


FUNCTION stop_leds()
INT i;
INT j;
INT mx_val;
INT poz;

lp1[1]=1;
lp1[2]=1;

FOR i=1 TO 10 DO
    mx_val=0;
    FOR j=1 TO 10 DO
        IF val_buttons[j]>=mx_val AND leduri[j]=1 THEN
           mx_val=val_buttons[j];
           poz=j;
        END
    END
    leduri[poz]=0;
    Sleep(1);
END
```